ELSEVIER

www.ElsevierComputerScience.com

POWERED BY SCIENCE DIRECT®

Computational Geometry 27 (2004) 151–178

Computational Geometry
Theory and Applications

www.elsevier.com/locate/comgeo

Inner and outer approximations of polytopes using boxes

Alberto Bemporad [a,*], Carlo Filippi [b], Fabio D. Torrisi [c]

[a] *Dipartimento di Ingegneria dell'Informazione, Università di Siena, Via Roma 56, 53100 Siena, Italy*
[b] *Dipartimento di Matematica Pura e Applicata, Università di Padova, Via Belzoni 7, 35131 Padova, Italy*
[c] *Automatic Control Laboratory, ETH Zentrum, ETL K13.2, 8092 Zurich, Switzerland*

Received 3 December 2002; received in revised form 2 June 2003; accepted 10 July 2003

Communicated by K. Mehlhorn

Abstract

This paper deals with the problem of approximating a convex polytope in any finite dimension by a collection of (hyper)boxes. More exactly, given a polytope $\mathcal{P}$ by a system of linear inequalities, we look for two collections $\mathcal{I}$ and $\mathcal{E}$ of boxes with non-overlapping interiors such that the union of all boxes in $\mathcal{I}$ is contained in $\mathcal{P}$ and the union of all boxes in $\mathcal{E}$ contains $\mathcal{P}$. We propose and test several techniques to construct $\mathcal{I}$ and $\mathcal{E}$ aimed at getting a good balance between two contrasting objectives: minimize the volume error and minimize the total number of generated boxes. We suggest how to modify the proposed techniques in order to approximate the projection of $\mathcal{P}$ onto a given subspace without computing the projection explicitly.

Keywords: Polytopes; Approximation; Boxes; Containment; Reachability analysis

1. Introduction

In this paper we formalize and solve the following problem in computational geometry. Given a full-dimensional convex polytope $\mathcal{P} \subset \mathbb{R}^d$, find two collections $\mathcal{I}$ and $\mathcal{E}$ of full-dimensional boxes such that: (i) the intersection between any two boxes is not full-dimensional; (ii) the intersection between any box in collection $\mathcal{E}$ and $\mathcal{P}$ is full-dimensional; (iii) the union of all boxes in $\mathcal{I}$ is contained in $\mathcal{P}$; (iv) the union of all boxes in $\mathcal{E}$ contains $\mathcal{P}$. Under the above properties, we say that the collection of boxes $\mathcal{I}$ is an *inner approximation* for the polytope $\mathcal{P}$, whereas the collection $\mathcal{E}$ is an *outer approximation* of $\mathcal{P}$, see Fig. 1. This

* Corresponding author.
E-mail addresses: bemporad@unisi.it (A. Bemporad), carlo@math.unipd.it (C. Filippi), torrisi@aut.ee.ethz.ch (F.D. Torrisi).

0925-7721/$ – see front matter
doi:10.1016/S0925-7721(03)00048-8

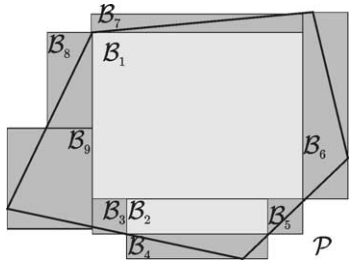


Fig. 1. Inner $\mathcal{I} = \{\mathcal{B}_i\}_{i=1}^{2}$ and outer $\mathcal{E} = \{\mathcal{B}_i\}_{i=1}^{9}$ approximations of a polytope $\mathcal{P}$.

formulation of the problem assumes that a *convex polyhedron* is the intersection of a finite set of closed halfspaces of the Euclidean space $\mathbb{R}^d$, and a *convex polytope* is a bounded convex polyhedron. Convex polytopes are important objects in applied sciences and computational techniques, and are often the key tools to solve problems in mathematical programming, computational geometry, statistics or control engineering [2,14,15,17]. A *box* is a convex polytope where all the defining hyperplanes are axis parallel.

In order to asses the performance of the approximation, we consider two quality indicators: *volume error*, defined as $(\mathrm{vol}(\mathcal{P}) - \mathrm{vol}(\mathcal{I}))/\mathrm{vol}(\mathcal{P})$ for an inner approximation, $(\mathrm{vol}(\mathcal{E}) - \mathrm{vol}(\mathcal{P}))/\mathrm{vol}(\mathcal{P})$ for an outer approximation, where the volume of a collection is the volume of the union of its items, and *cardinality*, defined as the number of boxes used in the inner and the outer approximation respectively. In principle, both the above indicators should be minimized, although clearly the objectives of having both volume error and cardinality small are contrasting. Our aim is then to obtain a good balance between them.

The motivation of the present work comes from the problem of reachability analysis of hybrid dynamical systems [15], namely dynamical processes of a heterogeneous continuous and discrete nature that switch among many operating modes, where each mode is governed by its own characteristic dynamical laws (difference or differential equations). Reachability analysis aims to answer questions like "will ever a hybrid dynamical system enter a critical region of operation?" or "will its quantities always be within a safe set?" by computing the set of configurations that the system can reach. The computation of "reach sets" amounts to perform a sequence of Minkowsky sums, deformations, intersections and projections of polyhedral sets. Although exact computation is possible [7,8], the complexity of the reachability analysis is reduced by replacing the complex original polyhedron with an inner (outer) approximation made of boxes. The fact that the approximation is strictly inner (outer) is of critical importance to conservatively answer the questions that reachability analysis poses. A small volume error keeps the conservativeness of the answer low, while the fact that the approximation has a small cardinality is important for computation efficiency (see [15] for details).

Similar approximation problems have been faced in [4,16]. In [16] the author presents an algorithm to approximate a convex polyhedron with one axis-parallel box, which is neither an inner nor an outer approximation. In [4] the authors discuss the problem of computing a lower bound to the volume of a polytope by adaptively filling the polytope with hypercubes. In this case the only objective is to minimize the volume error, therefore the number of boxes is not minimized (for a survey on exact volume computation of polytopes see [3]).

In our context, finding a single maximum volume box contained in a given polytope is a crucial subproblem, that may be classified as a "containment problem". The complexity of some general containment problems related to polyhedra is studied in [5,6]. In particular, in [5] it is shown that the problem of computing a maximum scaling of a polytope $\mathcal{P}$ such that its translation is contained in a given

polytope $\mathcal{Q}$ is a linear program. The same problem can be seen as the computation of the "inradius" of $\mathcal{Q}$ with respect to the polytopal norm induced by $\mathcal{P}$. This and several related problems concerning inner and outer radii of polytopes in finite-dimensional normed spaces were examined in detail in [10]. The relation between our results and those presented in [10] will be pointed out later. A comprehensive survey on containment problems can be found in [11].

In this paper, we aim at minimizing both the volume error introduced by the approximation and the cardinality of the approximation (number of boxes). The key idea is to proceed recursively: first we approximate the polytope $\mathcal{P}$ with one box $\mathcal{B}_1$ (as depicted in Fig. 1), then we partition the part that is not covered ($\mathcal{P} \setminus \mathcal{B}_1$) into polytopes having non-overlapping interiors, and then proceed iteratively in each obtained polytope. The advantage of this approach is to separate the two objectives: at each recursion we minimize the error, while the number of boxes (and therefore the complexity of the approximation) is limited by the number of recursions. After providing the necessary preliminaries, in Section 2 we detail different techniques to compute a single box inner and outer approximation. In Section 3 the basic recursive scheme is presented and used to formulate the inner, outer and inner-outer recursive approximation algorithms, and to analyze their behavior. Section 4 reports some computational experiences for the different algorithms. In Section 5 we show how the algorithm can be used to compute approximate projections of a polytope over an affine subspace. Section 6 concludes the paper stating some directions for future research.

Before proceeding further, we give some notation and recall some general definitions [17]. We represent a convex polyhedron $\mathcal{P} \in \mathbb{R}^d$ as $\mathcal{P} = \{x \in \mathbb{R}^d\colon\ Ax \leqslant b\}$, where A is a real $m \times d$ matrix and b is a real m-vector. An *interior point* of $\mathcal{P}$ is a point $\hat{x} \in \mathbb{R}^d$ such that $A\hat{x} < b$. A polyhedron $\mathcal{P}$ is *full-dimensional* if $\mathcal{P}$ has an interior point; otherwise, if it is embedded in a lower dimensional affine space, $\mathcal{P}$ is called *flat*.

Let $\mathcal{P}$ be a full-dimensional convex polytope in $\mathbb{R}^d$. The *faces* of $\mathcal{P}$ are the sets of form $\mathcal{P} \cap \{x \in \mathbb{R}^d\colon\ a'x = b\}$ for some valid inequality $a'x \leqslant b$. We say the face $\mathcal{P} \cap \{x \in \mathbb{R}^d\colon\ a'x = b\}$ is *determined by the inequality* $a'x \leqslant b$. The faces of dimension 0, 1 and $d-1$ are called *vertices*, *edges* and *facets*, respectively. A valid inequality $a'x \leqslant b$ is said to be a *facet inequality* if it determines a facet.

We represent a box as $\mathcal{B}(l,u) = \{x \in \mathbb{R}^d\colon\ l \leqslant x \leqslant u\}$, where l and u are real d-vectors. Note that $\mathcal{B}(l,u)$ is nonempty if and only if $l \leqslant u$ and it is full-dimensional if and only if $l < u$. Two full-dimensional boxes are *overlapping* if their intersection is a full-dimensional box. A *hypercube* is a box $\mathcal{B}(l,u)$ such that $u = l + \lambda e$, where λ is a scalar and e denotes the d-vector of all ones. Let $e_j \in \mathbb{R}^d$ be the jth column of the $d \times d$ identity matrix, $j = 1, \ldots, d$.

Let $\mathcal{C} \subseteq \mathbb{R}^d$ be a bounded, compact and closed set. Then the *volume* of $\mathcal{C}$, $\mathrm{vol}(\mathcal{C}) = \int_{\mathcal{C}} \mathrm{d}x$ is the Lebesgue measure of $\mathcal{C}$. A polytope $\mathcal{P}$ is full-dimensional if and only if it has a positive volume. Finally, let $D \triangleq \{1, 2, \ldots, d\}$ and $M \triangleq \{1, 2, \ldots, m\}$.

Most of the methods proposed in this paper are based on the solution of auxiliary linear programs. Since the time complexity of solving a linear program depends on the adopted solver (e.g., interior-point methods, simplex methods, randomized methods [9, Chapter 39]), we consider a linear program to be an oracle and evaluate the complexity of a given algorithm by the maximum number of linear programs that must be solved. More precisely, we denote by $\mathbf{lp}(m,d)$ the time complexity for solving an $m \times d$ canonical linear program $\max_x\{c^{\mathrm{T}}x\colon\ Ax \leqslant b\}$ where $A \in \mathbb{R}^{m\times d}$, $c \in \mathbb{R}^d$ and $b \in \mathbb{R}^m$. Since its dual linear program is a $d \times m$ canonical linear program, we may assume that $\mathbf{lp}(d,m)$ is of the same order as $\mathbf{lp}(m,d)$.

2. Single box approximation

Given a polyhedron $\mathcal{P} = \{x \in \mathbb{R}^d\colon\ Ax \leqslant b\}$ and a box $\mathcal{B}(l,u)$, we say that $\mathcal{B}(l,u)$ is an *inner* box of $\mathcal{P}$ if $\mathcal{B}(l,u) \subseteq \mathcal{P}$, $\mathcal{B}(l,u)$ is an *outer* box of $\mathcal{P}$ if $\mathcal{B}(l,u) \supseteq \mathcal{P}$. We address here the following problems:

(a) compute a maximum volume inner box of $\mathcal{P}$;
(b) compute a minimum volume outer box of $\mathcal{P}$.

We suggest an effective formulation for problem (a), which leads to a polynomial time solution. However, we focus on two simpler problems, namely:

(a.1) compute a maximum volume inner box of $\mathcal{P}$ such that the ratios among the edge lengths are a priori fixed;
(a.2) compute an inner box of $\mathcal{P}$ that is maximal with respect to inclusion.

Note that problem (a.1) is equivalent to computing the "inradius" of $\mathcal{P}$ with respect to the polytopal norm induced by a box with fixed edge ratios, whereas problem (b) is similar to computing the "circumradius" of $\mathcal{P}$ with respect to the maximum-norm. Both these general problems were studied in detail in [10], where solutions based on linear programming are suggested. In the sequel, we characterize the linear programs associated with (a.1), and propose an algorithm for solving problem (a.2) in strongly polynomial time, provided a point of $\mathcal{P}$ is known. Finally, we show how to solve problem (b) efficiently by linear programming.

2.1. Maximum volume inner box

A box $\mathcal{B}(l,u)$ can be written as $\mathcal{B}(x, x+y)$ by setting $x = l$ and $y = u - l$. Then, $\mathrm{vol}(\mathcal{B}(x, x+y)) = \prod_{j\in D} y_j$. Let $v(S) \in \{0,1\}^d$ be the incidence vector of the subset of coordinate indices $S \subseteq D$:

$$v_j(S) = \begin{cases} 1 & \text{if } j \in S, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Let $V(S) = \mathrm{diag}(v(S))$. The vertex set of $\mathcal{B}(x, x+y)$ may be expressed as $\{x + V(S)y\colon\ S \subseteq D\}$. By imposing that each vertex of $\mathcal{B}(x, x+y)$ is contained in $\mathcal{P}$, we formulate the following optimization problem:

$$\begin{aligned} \max_{x,y} \quad & \prod_{j\in D} y_j \\ \text{subject to} \quad & Ax + AV(S)y \leqslant b \quad (\forall S \subseteq D) \\ & y \geqslant 0 \end{aligned} \tag{2}$$

By construction, an optimal solution (x^*, y^*) of problem (2) identifies a maximum volume inner box $\mathcal{B}(x^*, x^* + y^*)$. Note that this intuitive formulation has a strongly nonlinear objective function and $m2^d$ linear constraints. Next Lemma 1 and Proposition 1 show that problem (2) is equivalent to the maximization of a concave function subject to m linear constraints.

Lemma 1. *The constraints in* (2) *are equivalent to the set of constraints* $Ax + A^+y \leqslant b$, *where* A^+ *is the positive part of* A, *namely* $a^+_{ij} = \max(0, a_{ij})$.

Proof. The constraints in (2) corresponding to any $i \in M$ can be written as

$$\sum_{j\in D} a_{ij}x_j + \sum_{j\in S} a_{ij}y_j \leqslant b_i \quad (\forall S \subseteq D),$$

i.e.,

$$\sum_{j\in S} a_{ij}y_j \leqslant b_i - \sum_{j\in D} a_{ij}x_j \quad (\forall S \subseteq D). \tag{3}$$

Let $S_i^+ = \{j\colon\ a_{ij} > 0\}$. Then condition $y > 0$ implies

$$\sum_{j\in S} a_{ij}y_j \leqslant \sum_{j\in S_i^+} a_{ij}y_j$$

for all $S \subseteq D$. It follows that all constraints (3) with $S \neq S_i^+$ are redundant and thus can be omitted. Using this fact, and noting that $\sum_{j\in S_i^+} a_{ij}y_j = \sum_{j\in D} a_{ij}^+ y_j$, the lemma is proved. □

Proposition 1. *Let $\mathcal{P} = \{x \in \mathbb{R}^d\colon\ Ax \leqslant b\}$ be a full-dimensional polytope, and let (x^*, y^*) be an optimal solution of*

$$\begin{aligned} &\max_{x,y} && \sum_{j\in D} \ln y_j \\ &\textit{subject to} && Ax + A^+y \leqslant b. \end{aligned} \tag{4}$$

Then $\mathcal{B}(x^*, x^* + y^*)$ is a maximum volume inner box of $\mathcal{P}$.

Proof. The result easily follows from Lemma 1, as $\mathcal{P}$ is full-dimensional, and therefore $y^* > 0$, and as the natural logarithm is a strictly monotonic function. □

According to Proposition 1, the maximum volume inner box can be computed as follows.

Algorithm 1.

function single-inner-nlp $(\mathcal{P})$
1 $A^+ = \max(A, 0)$;
2 solve (4);
3 **return** $\mathcal{B}(x^*, x^* + y^*)$.

By following the lines proposed in [13, Chapters 3 and 5], a path-following interior point method able to solve problem (4) within a polynomial number of Newton steps may be designed. Thus Algorithm 1 can be regarded as a polynomial time algorithm.

Since problem (4) will never be considered in the sequel, we omit here the (technical) formal proof of the above statement. On the contrary, in order to solve the single box approximation by a widely available and/or easily implementable code, we next consider two easier problems.

2.2. Maximum volume r-constrained inner box

Given a strictly positive vector $r \in \mathbb{R}^d$, we say that box $\mathcal{B}(l,u)$ is *r-constrained* if $u = l + \lambda r$ for some scalar $\lambda \geqslant 0$. We may interpret r as the vector of fixed edge length ratios. We can find a maximum volume r-constrained box contained in $\mathcal{P}$ by solving a linear program with $d+1$ variables and m constraints.

Proposition 2. *Let $\mathcal{P} = \{x \in \mathbb{R}^d\colon\ Ax \leqslant b\}$ be a nonempty polytope, and let (x^*, λ^*) be an optimal solution of*

$$\begin{aligned} &\max_{x,\lambda} && \lambda \\ &\text{subject to} && Ax + A^+ r\lambda \leqslant b, \end{aligned} \tag{5}$$

where $r \in \mathbb{R}^d$ is strictly positive and λ is a scalar, and A^+ is the positive part of A. Then $\mathcal{B}(x^*, x^* + \lambda^* r)$ is a maximum volume r-constrained inner box of $\mathcal{P} = \{x \in \mathbb{R}^d\colon\ Ax \leqslant b\}$.

Proof. We prove that problem (2) with the additional constraint $y = \lambda r$ is equivalent to problem (5). If we set $y = \lambda r$, where $r \in \mathbb{R}^d$ is a strictly positive fixed vector and λ is a scalar nonnegative variable, then the objective function of problem (2) becomes $(\prod_{j\in D} r_j)\lambda^d$ that, for nonnegative λ, is a strictly monotonic function of λ. Hence problem (2) with $y = \lambda r$ is equivalent to

$$\max_{x,\lambda}\big\{\lambda\colon\ Ax + AV(S)r\lambda \leqslant b\ (\forall S \subseteq D),\ \lambda \geqslant 0\big\}. \tag{6}$$

The result follows by applying Lemma 1 with $y = r\lambda$. □

Note that an optimal solution of problem (5) with strictly positive λ exists if and only if $\mathcal{P}$ is full-dimensional. Note further that problem (5) can have more than one optimal solution.

The choice of the edge length ratios r is crucial for the quality of the generated r-constrained inner box. Working with hypercubes (all edge length ratios equal to one) seems to be a reasonable choice when we have no information about the shape of $\mathcal{P}$, but may be very inefficient when the ratio between the "width" of $\mathcal{P}$ (i.e., the smallest distance between pairs of parallel supporting hyperplanes of $\mathcal{P}$) and the "diameter" of $\mathcal{P}$ (i.e., the largest distance realized between two points of $\mathcal{P}$) is comparatively small. A possible choice for the edge length ratio vector r is

$$r_j = \lambda_j(\mathcal{P}) \quad \text{for all } j \in D, \tag{7}$$

where $\lambda_j(\mathcal{P})$ denotes the maximum length of a line segment parallel to the jth coordinate axis and contained in $\mathcal{P}$.

Proposition 3. *Let $\mathcal{P} = \{x \in \mathbb{R}^d\colon\ Ax \leqslant b\}$ be a nonempty polytope. Then $\lambda_j(\mathcal{P}) = \max_{x,\lambda}\{\lambda\colon\ Ax + A_j^+\lambda \leqslant b\}$, where A_j^+ denotes the jth column of the positive part of A.*

Proof. A line segment parallel to the jth coordinate axis may be written as $E = \operatorname{conv}(x, x + \lambda e_j)$ for some $x \in \mathbb{R}^d$, where e_j denotes the jth column of the $d \times d$ identity matrix and λ is the length of E. As a consequence,

$$\lambda_j(\mathcal{P}) = \max_{x,\lambda}\big\{\lambda\colon\ Ax \leqslant b,\ A(x + \lambda e_j) \leqslant b,\ \lambda \geqslant 0\big\}.$$

The above linear program has two constraints for each $i \in M$, namely

$$\sum_{k\in D} a_{ik}x_k \leqslant b_i \quad \text{and} \quad \sum_{k\in D} a_{ik}x_k + a_{ik}\lambda \leqslant b_i.$$

Since $\lambda \geqslant 0$, if $a_{ik} \leqslant 0$ then the latter constraint is redundant; if $a_{ik} > 0$ then the former constraint is redundant. Hence both constraints may be replaced by $\sum_{k\in D} a_{ik}x_k + a_{ik}^+\lambda \leqslant b_i$. If $\mathcal{P}$ is not empty then constraint $\lambda \geqslant 0$ is redundant and thus can be omitted. The statement follows. □

The complexity of computing $\lambda_j(\mathcal{P})$ for all $j \in D$ is then $O(d\,\mathbf{lp}(m, d+1))$.

An alternative choice is to use the ratio $r = u - l$ of the outer box computed according to the following Section 2.4. This choice is reasonable when both an inner and outer approximation of the same polytope are sought. In fact, if we compute first the outer box then we get as a byproduct a good vector r for the successive r-constrained inner box computation.

Once an r vector is given, the r-constrained inner box computation is formalized by the following algorithm.

Algorithm 2.

function single-inner-lp($\mathcal{P}$, r)
1 $A^+ = \max(A, 0)$;
2 solve (5);
3 **return** $\mathcal{B}(x^*, x^* + \lambda^* r)$.

The time complexity of Algorithm 2 is $O(\mathbf{lp}(m, d+1))$.

2.3. Greedy inner box

A box $\mathcal{B} \subseteq \mathcal{P}$ is said to be a *greedy inner box* if $\mathcal{B}$ is maximal with respect to inclusion, i.e., if it does not exist a box $\widetilde{\mathcal{B}} \neq \mathcal{B}$ such that $\mathcal{B} \subseteq \widetilde{\mathcal{B}} \subseteq \mathcal{P}$. Let $\mathcal{P}^k \subseteq \mathbb{R}^d$ be a polytope containing the origin on $\mathbb{R}^d$. We first show that the maximum volume hypercube contained in $\mathcal{P}^k$ and centered in the origin can be found by applying a simple formula. Then, we show how to apply the formula iteratively in order to obtain a greedy inner box of $\mathcal{P}$.

Let $\mathcal{P}^k = \{x \in \mathbb{R}^d\colon\ A^k x \leqslant b^k\}$ be a polytope containing the origin, so that $b^k \geqslant 0$. Consider hypercubes centered in the origin with edge length 2τ, denoted as $\mathcal{B}(-\tau e, +\tau e)$. Finding the maximum volume hypercube contained in $\mathcal{P}^k$ and centered in the origin is equivalent to finding

$$\tau(\mathcal{P}^k) = \max\{\tau\colon\ \mathcal{B}(-\tau e, +\tau e) \subseteq \mathcal{P}^k\}. \tag{8}$$

The above maximization problem can be solved in closed form as follows.

Proposition 4. *Let $A^k = [a_{ij}^k]$ and for all $i \in M$ let*

$$\tau_i = \begin{cases} \dfrac{b_i^k}{\sum_{j\in D} |a_{ij}^k|} & \text{if } \sum_{j\in D} |a_{ij}^k| > 0, \\ +\infty & \text{otherwise}. \end{cases} \tag{9}$$

Then $\tau(\mathcal{P}^k) = \min\{\tau_i\colon\ i \in M\}$.

Proof. For any $i \in M$, consider the linear program

$$z_i(\tau) = \max\left\{ \sum_{j\in D} a_{ij}^k x_j \colon\ x \in \mathcal{B}(-\tau e, +\tau e) \right\},$$

which has the straightforward optimal solution

$$x_j^* = \begin{cases} -\tau & \text{if } a_{ij}^k < 0, \\ +\tau & \text{if } a_{ij}^k \geqslant 0, \end{cases}$$

with corresponding objective value $z_i(\tau) = \tau \sum_{j\in D} |a_{ij}^k|$.

The inequality $\sum_{j\in D} a_{ij}^k x_j \leqslant b_i^k$ holds for all points in $\mathcal{B}(-\tau e, +\tau e)$ if and only if $z_i(\tau) \leqslant b_i^k$, i.e., if and only if $\tau \leqslant \tau_i$, where τ_i is defined as in (9). It follows that $\mathcal{B}(-\tau e, +\tau e)$ is contained in $\mathcal{P}$ if and only if $\tau \leqslant \tau_i$ for all $i \in M$. Since $\mathcal{B}(-\tau e, +\tau e) \subseteq \mathcal{B}(-\tau_i e, +\tau_i e)$ if and only if $\tau \leqslant \tau_i$ and $\mathcal{B}(-\tau e, +\tau e) = \mathcal{B}(-\tau_i e, +\tau_i e)$ if and only if $\tau = \tau_i$, we conclude that $\tau(\mathcal{P}^k)$ is the minimum τ_i. □

The volume of the hypercube obtained by Proposition 4 greatly depends on the relative position of the origin inside $\mathcal{P}^k$ and in particular it is zero if the origin lies onto the boundary of $\mathcal{P}^k$. Nevertheless, Proposition 4 may be applied iteratively to obtain a greedy inner box. In order to do that, we start with a few observations.

Let $\tau_{\bar\imath} = \tau(\mathcal{P}^k)$ for some $\bar\imath \in M$. If $x_{\bar\jmath}$ has a negative coefficient in the $\bar\imath$th inequality of $A^k x \leqslant b^k$, then any box $\mathcal{B}(l,u)$ such that $\mathcal{B}(-\tau(\mathcal{P}^k)e, \tau(\mathcal{P}^k)e) \subseteq \mathcal{B}(l,u) \subseteq \mathcal{P}^k$ must have $l_{\bar\jmath} = -\tau(\mathcal{P}^k)$. Moreover, for all $i \in M$ such that $a_{i\bar\jmath}^k < 0$ we have

$$\begin{cases} \sum_{j\in D\setminus\{\bar\jmath\}} a_{ij}^k x_j + a_{i\bar\jmath}^k x_{\bar\jmath} \leqslant b_i \\ x_{\bar\jmath} \geqslant -\tau(\mathcal{P}^k) \end{cases} \quad\Leftrightarrow\quad \begin{cases} \sum_{j\in D\setminus\{\bar\jmath\}} a_{ij}^k x_j \leqslant b_i + a_{i\bar\jmath}^k \tau(\mathcal{P}^k) \\ x_{\bar\jmath} \geqslant -\tau(\mathcal{P}^k). \end{cases}$$

Symmetrically, if $x_{\bar\jmath}$ has a positive coefficient in the $\bar\imath$th inequality of $A^k x \leqslant b^k$ then any box $\mathcal{B}(l,u)$ such that $\mathcal{B}(-\tau(\mathcal{P}^k)e, \tau(\mathcal{P}^k)e) \subseteq \mathcal{B}(l,u) \subseteq \mathcal{P}^k$ must have $u_{\bar\jmath} = \tau(\mathcal{P}^k)$, and for all $i \in M$ such that $a_{i\bar\jmath}^k > 0$ we have

$$\begin{cases} \sum_{j\in D\setminus\{\bar\jmath\}} a_{ij}^k x_j + a_{i\bar\jmath}^k x_{\bar\jmath} \leqslant b_i \\ x_{\bar\jmath} \leqslant \tau(\mathcal{P}^k) \end{cases} \quad\Leftrightarrow\quad \begin{cases} \sum_{j\in D\setminus\{\bar\jmath\}} a_{ij}^k x_j \leqslant b_i - a_{i\bar\jmath}^k \tau(\mathcal{P}^k) \\ x_{\bar\jmath} \leqslant \tau(\mathcal{P}^k). \end{cases}$$

It follows that, once we fix the lower bound on one variable, this variable can be removed from all the inequalities defining $\mathcal{P}^k$ where it has negative coefficient. Symmetrically, once we fix the upper bound on one variable, this variable can be removed from all the inequalities defining $\mathcal{P}^k$ where it has positive coefficient. In this way, we may transform the system $A^k x \leqslant b^k$ in a new system $A^{k+1} x \leqslant b^{k+1}$ where the coefficient matrix has a strictly less number of nonzero coefficients. On this new system defining a (possibly unbounded) polyhedron $\mathcal{P}^{k+1}$ we may compute $\tau(\mathcal{P}^{k+1})$ and repeat the transformation.

We formalize the above argument as follows.

Consider the quintuple $\mathcal{Q}^k = \{\mathcal{P}^k, l^k, u^k, L^k, U^k\}$ where $\mathcal{P}^k = \{x \in \mathbb{R}^d \colon\ A^k x \leqslant b^k\}$, with $A^k = [a_{ij}^k]$ real $m \times d$ matrix and b^k real m-vector; l^k, u^k are real d-vectors; and L^k, U^k are index subsets, contained in D. We assume the following holds:

$$j \in L^k \quad \text{implies} \quad a_{ij}^k \geqslant 0 \text{ for all } i \in M, \tag{10a}$$

$$j \in U^k \quad \text{implies} \quad a_{ij}^k \leqslant 0 \text{ for all } i \in M. \tag{10b}$$

Assume we are interested in finding a greedy inner box of $\mathcal{P} = \{x \in \mathcal{R}^d\colon\ Ax \leqslant b\}$ and assume we have a point x^0 in $\mathcal{P}$. By mapping x^0 onto the origin with the translation $x \to x - x^0$, we translate $\mathcal{P}$ to $\mathcal{P}^0 = \{x \in \mathbb{R}^d\colon\ A^0x \leqslant b^0\}$, where $A^0 = A$ and $b^0 = b - Ax^0$, with $b^0 \geqslant 0$. We define $\mathcal{Q}^0 = (\mathcal{P}^0, l^0, u^0, L^0, U^0)$ where

$$\mathcal{P}^0 = \{x \in \mathbb{R}^d\colon\ A^0x \leqslant b^0\}, \qquad l^0 = u^0 = 0, \qquad L^0 = U^0 = \emptyset. \tag{11}$$

Note that property (10) trivially holds for $\mathcal{Q}^0$.

Given a generic $\mathcal{Q}^k$, compute $\tau(\mathcal{P}^k)$ by using Proposition 4 and define $M^k = \{i \in M\colon\ \tau_i = \tau(\mathcal{P}^k)\}$. For all $i \in M$ define

$$L_i^k = \{j \in D\colon\ a_{ij}^k < 0 \text{ and } a_{hj}^k < 0 \text{ for some } h \in M^k\}, \tag{12a}$$

$$U_i^k = \{j \in D\colon\ a_{ij}^k > 0 \text{ and } a_{hj}^k > 0 \text{ for some } h \in M^k\}. \tag{12b}$$

Clearly, $L_i^k \cap U_i^k = \emptyset$. Note that if $i \in M^k$ then L_i^k and U_i^k are the index sets of respectively negative and positive entries in the ith inequality of system $A^kx \leqslant b^k$. Note further that $\bigcup_{i\in M^k} L_i^k = \bigcup_{i\in M} L_i^k$ and $\bigcup_{i\in M^k} U_i^k = \bigcup_{i\in M} U_i^k$.

Apply the following iterative transformations:

$$a_{ij}^{k+1} = \begin{cases} 0 & \text{if } j \in L_i^k \cup U_i^k, \\ a_{ij}^k & \text{otherwise,} \end{cases} \quad (i \in M,\ j \in D), \tag{13a}$$

$$b_i^{k+1} = b_i^k - \tau(\mathcal{P}^k)\left(\sum_{j\in L_i^k} |a_{ij}^k| + \sum_{j\in U_i^k} |a_{ij}^k|\right) \quad (i \in M), \tag{13b}$$

$$l_j^{k+1} = \begin{cases} -\tau(\mathcal{P}^k) & \text{if } j \in \bigcup_{i\in M} L_i^k, \\ l_j^k & \text{otherwise,} \end{cases} \quad (j \in D), \tag{13c}$$

$$u_j^{k+1} = \begin{cases} +\tau(\mathcal{P}^k) & \text{if } j \in \bigcup_{i\in M} U_i^k, \\ u_j^k & \text{otherwise,} \end{cases} \quad (j \in D), \tag{13d}$$

$$L^{k+1} = L^k \cup \left(\bigcup_{i\in M} L_i^k\right), \tag{13e}$$

$$U^{k+1} = U^k \cup \left(\bigcup_{i\in M} U_i^k\right). \tag{13f}$$

Now define $\mathcal{P}^{k+1} = \{x \in \mathbb{R}^d\colon\ A^{k+1}x \leqslant b^{k+1}\}$ and $\mathcal{Q}^{k+1} = \{\mathcal{P}^{k+1}, l^{k+1}, u^{k+1}, L^{k+1}, U^{k+1}\}$. The properties of the above iterative transformations are summarized by the following result.

Theorem 1. *Starting from* $\mathcal{Q}^0$ *defined by* (11)*, apply the iterative transformations* (13) *until* $L^{k+1} = U^{k+1} = D$*. Then,* $\mathcal{B}(x^0 + l^{k+1}, x^0 + u^{k+1})$ *is a greedy inner box for* $\mathcal{P}$*.*

Proof. We prove that $\mathcal{B}(l^{k+1}, u^{k+1})$ is a greedy inner box for $\mathcal{P}^0$. We first prove that $\mathcal{B}(l^{k+1}, u^{k+1}) \subseteq \mathcal{P}^0$. Property (10) holds for $\mathcal{Q}^0$ and the iterative transformations (13) preserve (10). Let $i \in M$ be fixed. By applying repeatedly the iterative transformation (13b) we get

$$b_i^{k+1} = b_i^0 - \sum_{p=0}^{k} \tau(\mathcal{P}^p)\left(\sum_{j\in L_i^p} |a_{ij}^p| + \sum_{j\in U_i^p} |a_{ij}^p| \right).$$

By the iterative transformations (13c)–(13d), for any $\mathcal{P}$ if $j \in L_i^p$ then $l_j^{p+1} = -\tau(\mathcal{P}^p)$, if $j \in U_i^p$ then $u_j^{p+1} = +\tau(\mathcal{P}^p)$. Thus we get

$$b_i^{k+1} = b_i^0 - \left(\sum_{p=0}^{k} \sum_{j\in L_i^p} a_{ij}^p l_j^{p+1} + \sum_{p=0}^{k} \sum_{j\in U_i^p} a_{ij}^p u_j^{p+1} \right). \tag{14}$$

Property (10), definitions (12), and the iterative transformation (13a) imply $L_i^p \cap L_i^q = \emptyset$ and $U_i^p \cap U_i^q = \emptyset$ for all $q \neq p$. This fact has the following consequences: (i) in (14) every index j appears at most once in each summation; (ii) by the iterative transformations (13c)–(13d) if $j \in L_i^p$ (respectively $j \in U_i^p$) for some $0 \leqslant p \leqslant k$ then $l_j^{p+1} = l_j^{k+1}$ (respectively $u_j^{p+1} = u_j^{k+1}$); (iii) by the iterative transformation (13a) if $j \in L_i^p$ or $j \in U_i^p$ then $a_{ij}^{p+1} = 0$ but $a_{ij}^p = a_{ij}^0$. Thus (14) is equivalent to

$$b_i^{k+1} = b_i^0 - \left(\sum_{j\in L_i} a_{ij}^0 l_j^{k+1} + \sum_{j\in U_i} a_{ij}^0 u_j^{k+1} \right), \tag{15}$$

where $L_i = \bigcup_{p=0}^{k} L_i^p$ and $U_i = \bigcup_{p=0}^{k} U_i^p$. By definition (12) we have $L_i^p \cap L_i^q = \emptyset$ for all p, q and hence $L_i \cap U_i = \emptyset$. By property (10), if $L^{k+1} = U^{k+1} = D$ then necessarily $a_{ij}^{k+1} = 0$ for all i and j; for all $j \in L_i \cup U_i$, the iterative transformation (13a) implies $a_{ij}^{k+1} = a_{ij}^0$ and hence $a_{ij}^0 = 0$. Thus, for all $\bar{x} \in \mathcal{B}(l^{k+1}, u^{k+1})$,

$$\sum_{j\in D} a_{ij}^0 \bar{x}_j = \sum_{j\in L_i} a_{ij}^0 \bar{x}_j + \sum_{j\in U_i} a_{ij}^0 \bar{x}_j \leqslant \sum_{j\in L_i} a_{ij}^0 l_j^{k+1} + \sum_{j\in U_i} a_{ij}^0 u_j^{k+1} = b_i^0 - b_i^{k+1}. \tag{16}$$

Observe that if $i \in M^k$ then $b_i^{k+1} = 0$; if $i \notin M^k$ then $b_i^{k+1} \geqslant b_i^k - \tau(\mathcal{P}^k) \sum_{j\in D} |a_{ij}^k| > b_i^k - b_i^k = 0$. Hence (16) implies $\sum_{j\in D} a_{ij}^0 \bar{x}_j \leqslant b_i^0$. Since this is true for all $i \in M$, then $\mathcal{B}(l^{k+1}, u^{k+1}) \subseteq \mathcal{P}^0$.

Next suppose that $\mathcal{B}(l^{k+1}, u^{k+1})$ is not greedy. Suppose $\mathcal{B}(l^{k+1}, u^{k+1}) \subset \mathcal{B}(\bar{l}, \bar{u}) \subseteq \mathcal{P}^0$ for some $\bar{l}$ and $\bar{u}$. Then $\bar{l} \leqslant l^{k+1}$, $\bar{u} \geqslant u^{k+1}$, and there must be an index $h \in D$ such that (i) $\bar{l}_h < l_h^{k+1}$ or (ii) $\bar{u}_h > u_h^{k+1}$. Assume that (i) holds. Let l_h^{k+1} be fixed during the qth iteration, for some $0 \leqslant q \leqslant k$, and let $i \in M^q$. Let now $L_i = \bigcup_{p=0}^{q} L_i^p$ and $U_i = \bigcup_{p=0}^{q} U_i^p$, and let $\bar{x} = [\bar{x}_j]$ where

$$\bar{x}_j = \begin{cases} \bar{l}_j & \text{if } j \in L_i, \\ \bar{u}_j & \text{if } j \in U_i, \\ 0 & \text{otherwise.} \end{cases}$$

Clearly, $\bar{x} \in \mathcal{B}(\bar{l}, \bar{u})$. By construction, $b_i^{q+1} = 0$ and so $b_i^{k+1} = 0$. Hence we obtain

$$\sum_{j\in D} a_{ij}^0 \bar{x}_j = \sum_{j\in L_i} a_{ij}^0 \bar{l}_j + \sum_{j\in U_i} a_{ij}^0 \bar{u}_j > \sum_{j\in L_i} a_{ij}^0 l_j^{k+1} + \sum_{j\in U_i} a_{ij}^0 u_j^{k+1} = b_i^0.$$

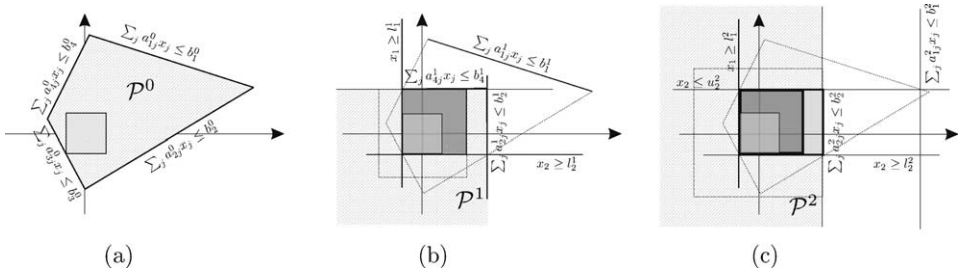


Fig. 2. Recursive application of (13) to produce a greedy inner box in a 2-dimensional example. The polytopes $\mathcal{P}^k$ are the shaded part of the plane. (a) Since $\tau(\mathcal{P}^0) = \tau_3$ and $a^0_{3j} < 0$ for $j = 1, 2$, we fix $l^1_1 = l^1_2 = -\tau(\mathcal{P}^0)$. We then remove all negative coefficients in matrix A^0, getting A^1, and we update b^0, getting b^1. Note that the first row of A^0 have positive entries, so that the corresponding inequality does not change. (b) Since $\tau(\mathcal{P}^1) = \tau_4$ and $a^0_{4,2} > 0$, we fix $u^2_2 = \tau(\mathcal{P}^1)$. Now, the second inequality remains unchanged. (c) Polyhedron $\mathcal{P}^2$ is simply a halfplane. Since $\tau(\mathcal{P}^2) = \tau_2$ and $a^2_{2,1} > 0$, we fix $u^3_1 = \tau(\mathcal{P}^2)$. Now all variables have both a lower and an upper bound and the procedure stops returning the bold inner box, which is greedy for the original polytope $\mathcal{P}^0$.

This implies that $\bar{x} \notin \mathcal{P}^0$, making a contradiction. If (ii) holds then we can proceed analogously to get a contradiction. □

Theorem 1 guarantees that the recursive application of (13) produces a greedy inner box. The idea is depicted in Fig. 2, and is formally summarized by the following algorithm.

Algorithm 3.

function single-inner-greedy($\mathcal{P}$, x_0)
1 $k = 0$;
2 $A^k = A$; $b^k = b - Ax_0$; $l^k = 0$; $u^k = 0$; $L^k = \emptyset$; $U^k = \emptyset$;
3 **while** $L^k \neq D$ or $U^k \neq D$,
4 compute $\tau(\mathcal{P}^k)$ by using Proposition 4;
5 apply the iterative transformations (13);
6 $k = k + 1$;
7 return $\mathcal{B}(l^k, u^k)$.

Proposition 5. *Algorithm* 3 *runs in* $O(md^2)$ *time.*

Proof. At the end of each while loop at least one new index is added to L^k or U^k. Hence, at most $2d$ while loops are performed. Each while loop consists of the computation of $\tau(\mathcal{P}^k)$ and the application of the iterative transformations (13). Both these operations require $O(md)$ time. □

Note that Algorithm 3 is strongly polynomial provided that a point $x_0 \in \mathcal{P}$ is given.

2.4. Minimum volume outer box

The problem of finding the minimum volume outer box containing $\mathcal{P}$ amounts to solve $2d$ linear programs with d variables and m constraints. Let

$$l_j = \min\{x_j\colon\ Ax \leqslant b\}, \tag{17a}$$

$$u_j = \max\{x_j\colon\ Ax \leqslant b\}, \tag{17b}$$

for all $j \in D$. By construction, $\mathcal{B}(l,u)$ is contained in every box containing $\mathcal{P}$. Then, $\mathcal{B}(l,u)$ is the unique minimum volume outer box of $\mathcal{P}$.

Algorithm 4.

function single-outer($\mathcal{P}$)
1 **for** $j = 1 \ldots d$,
2 $l_j = \min\{x_j\colon\ Ax \leqslant b\}$;
3 $u_j = \max\{x_j\colon\ Ax \leqslant b\}$;
4 **return** $\mathcal{B}(l,u)$.

Clearly, the time complexity of Algorithm 4 is $O(d\mathbf{lp}(m,d))$.

3. Multiple box approximation

Let us summarize the results obtained so far: the problem of finding the largest hyper-rectangular inner approximation of a polytope $\mathcal{P} = \{x \in \mathbb{R}^d\colon\ Ax \leqslant b\}$ was cast as the convex nonlinear program (4). We also proposed two simpler suboptimal methods: Algorithm 2, based on r-constrained boxes, and Algorithm 3, based on greedy inner boxes. The algorithms exhibit a polynomial respectively a strongly polynomial complexity (provided a point of $\mathcal{P}$ in known). We suggested three different ways to determine vector r in Algorithm 2. The problem of finding an outer approximation was solved straightforwardly by Algorithm 4 in polynomial time.

The single-box inner approximation algorithms developed in the previous section can be applied recursively to obtain an inner approximation of $\mathcal{P}$ as the union of full-dimensional, non-overlapping boxes. The key idea is the following (cf. Fig. 1): Let $\mathcal{B}_1$ be an inner approximation of $\mathcal{P}$. Partition (the closure of) $\mathcal{P} \setminus \mathcal{B}_1$ into convex polytopes $\mathcal{P}_h$ ($h = 1, 2, \ldots$), and recursively compute an inner approximation $\mathcal{B}_{h+1}$ for each $\mathcal{P}_h$. This approximation technique is associated with a tree. Each inner approximation induces a partition of the remaining polytope and a branching. The polytopes $\mathcal{P}_h$ are further inner approximated until some termination condition is met. We will also consider a recursive approximation algorithm that directly computes an outer approximation.

The termination conditions are related to the accuracy of the approximation. Ideally, one would like to stop the recursive algorithm when the volume error induced by the approximation is smaller than a given tolerance ε ($\mathrm{vol}(\mathcal{P}) - \mathrm{vol}(\mathcal{I}) < \varepsilon \cdot \mathrm{vol}(\mathcal{P})$). This however is not practical as determining the exact volume of a polytope is computationally expensive [3].

An alternative is to stop the recursion when the volume of the outer approximation and the volume of the inner one are within a given interval ($\mathrm{vol}(\mathcal{E}) - \mathrm{vol}(\mathcal{I}) < \varepsilon \cdot \mathrm{vol}(\mathcal{E})$). However, such a criterion is applicable only when we are interested in both an inner and an outer approximation.

A better stopping condition is to prune a branch of the approximation tree if the generated inner box $\mathcal{B}$ is smaller than a given tolerance ($\mathrm{vol}(\mathcal{B}) < \varepsilon$). Note that this stopping criterion does not guarantee a direct bound on the volume error, although it is very simple to apply and it is justified by the asymptotic results given below (Theorems 2 and 3) and by the numerical results of Section 4.

3.1. Partitioning $\mathcal{P} \setminus \mathcal{B}(l,u)$

Given a box $\mathcal{B}(l,u) \subset \mathcal{R}^d$, for all $k \in D$ define the following pair of polyhedra:

$$\mathcal{H}_{2k-1} = \{x \in \mathbb{R}^d\colon\ l_j \leqslant x_j \leqslant u_j\ (1 \leqslant j \leqslant k-1), x_k \leqslant l_k\}, \tag{18}$$

$$\mathcal{H}_{2k} = \{x \in \mathbb{R}^d\colon\ l_j \leqslant x_j \leqslant u_j\ (1 \leqslant j \leqslant k-1), x_k \geqslant u_k\}. \tag{19}$$

Proposition 6. *We have* $\mathcal{B}(l,u) \cup (\bigcup_{r=1}^{2d} \mathcal{H}_r) = \mathbb{R}^d$.

Proof. Let $\hat{x} \in \mathbb{R}^d$. If $\hat{x} \in \mathcal{B}(l,u)$, the claim is trivially true. Otherwise, let $k \in D$ be the first index such that $\hat{x}_k \notin [l_k, u_k]$. If $\hat{x}_k < l_k$, then $\hat{x} \in \mathcal{H}_{2k-1}$, otherwise $\hat{x} \in \mathcal{H}_{2k}$. □

Proposition 7. *Let $\mathcal{P}$ be a full-dimensional polytope and let $\mathcal{B}(l,u)$ be a full-dimensional box contained in $\mathcal{P}$. The following claims hold*:

(1) $\dim(\mathcal{H}_h) = d$ *for all* $h = 1, \ldots, 2d$,
(2) $\dim(\mathcal{H}_h \cap \mathcal{H}_s) < d$ *for all* $h, s = 1, \ldots, 2d$ *with* $h \neq s$.

Proof. Recall that $\mathcal{B}(l,u)$ is full-dimensional if and only if $l < u$.

(1) The vectors $\hat{x}^{2k-1} = [\hat{x}_j^{2k-1}]$ and $\hat{x}^{2k} = [\hat{x}_j^{2k}]$ defined as

$$\hat{x}_j^{2k-1} = \begin{cases} \frac{1}{2}(u_j + l_j) & \text{if } j < k, \\ l_k - 1 & \text{if } j = k, \\ 0 & \text{otherwise,} \end{cases} \qquad \hat{x}_j^{2k} = \begin{cases} \frac{1}{2}(u_j + l_j) & \text{if } j < k, \\ u_k + 1 & \text{if } j = k, \\ 0 & \text{otherwise} \end{cases} \tag{20}$$

are interior points of $\mathcal{H}_{2k-1}$ and $\mathcal{H}_{2k}$, respectively.

(2) Assume $h < s$. If $h = 2k-1$ for some $k \in D$ then the description of $\mathcal{H}_h$ contains the inequality $x_k \leqslant l_k$, whereas the description of $\mathcal{H}_s$ contains either the inequality $x_k \geqslant u_k (> l_k)$ or the inequality $x_k \geqslant l_k$. In the former case, $\mathcal{H}_h \cap \mathcal{H}_s = \emptyset$; in the latter case, $\mathcal{H}_h \cap \mathcal{H}_s \subseteq \{x \in \mathbb{R}^d\colon\ x_k = l_k\}$. If $h = 2k$ for some $k \in D$ then the description of $\mathcal{H}_h$ contains the inequality $x_k \geqslant u_k$, whereas the description of $\mathcal{H}_s$ contains the inequality $x_k \leqslant u_k$, so that $\mathcal{H}_h \cap \mathcal{H}_s \subseteq \{x \in \mathbb{R}^d\colon\ x_k = u_k\}$. □

Define $\mathcal{P}_h = \mathcal{P} \cap \mathcal{H}_h$ for all $h = 1, \ldots, 2d$ and note that $\bigcup_{h=1}^{2d} \mathcal{P}_h$ equals the closure of $\mathcal{P} \setminus \mathcal{B}(l,u)$. Let $x_j^c = (u_j + l_j)/2$ be the coordinates of the center of $\mathcal{B}(l,u)$, $j \in D$. For all $k \in D$ let

$$\begin{aligned} \lambda_{2k-1} &= \max\left\{ \frac{b_i - \sum_{j \in D} a_{ij} x_j^c}{a_{ik}}\colon\ i \in M, a_{ik} < 0 \right\} + \frac{u_k - l_k}{2}, \\ \lambda_{2k} &= \min\left\{ \frac{b_i - \sum_{j \in D} a_{ij} x_j^c}{a_{ik}}\colon\ i \in M, a_{ik} > 0 \right\} - \frac{u_k - l_k}{2}. \end{aligned} \tag{21}$$

Proposition 8. *For all $k \in D$ the following holds*:

(1) $\mathcal{P}_{2k-1}$ *is full-dimensional if and only if* $\lambda_{2k-1} < 0$,
(2) $\mathcal{P}_{2k}$ *is full-dimensional if and only if* $\lambda_{2k} > 0$,
(3) $\dim(\mathcal{P}_h \cap \mathcal{P}_s) < d$ *for all* $h, s = 1, \ldots, 2d$ *with* $h \neq s$.

Proof. (1) Let $\mathcal{L} = \{x \in \mathbb{R}^d\colon\ x_k = l_k\}$ and consider the following inclusions:

$$(\mathcal{B}(l,u) \cap \mathcal{L}) \subseteq (\mathcal{P}_{2k-1} \cap \mathcal{L}) \subseteq \mathcal{P}_{2k-1}. \tag{22}$$

The set on the left of (22) has dimension $d-1$ and the set in the middle is a face of $\mathcal{P}_{2k-1}$. Thus, the dimension of $\mathcal{P}_{2k-1}$ is either d or $d-1$ and $\mathcal{P}_{2k-1} \cap \mathcal{L}$ is either a facet of $\mathcal{P}_{2k-1}$ or $\mathcal{P}_{2k-1}$ itself. Let $e^k \in \mathbb{R}^d$ be the kth column of the identity matrix. Point $l_k e^k$ lies on the relative interior of $\mathcal{P}_{2k-1} \cap \mathcal{L}$ and the direction e^k is orthogonal to $\mathcal{L}$. Thus, $\mathcal{P}_{2k-1}$ is full-dimensional if and only if it is possible to move from $l_k e^k$ along the direction e^k, i.e., if and only if $\lambda_{2k-1} < l_k$. The proof of item (2) is similar. Item (3) follows from $\dim(\mathcal{P}_h \cap \mathcal{P}_s) \leqslant \dim(\mathcal{H}_h \cap \mathcal{H}_s)$ and item (2) of Proposition 7. □

3.2. A recursive algorithm for inner approximation

The following Algorithm 5 computes an inner approximation of a full-dimensional polytope as the union of non-overlapping full-dimensional boxes.

Algorithm 5.

function $\mathcal{I} = \text{multi-inner}(\mathcal{P})$
1 $\mathcal{B} = \text{single-inner}(\mathcal{P})$;
2 **if** $\text{vol}(\mathcal{B}) > \varepsilon$,
3 $\mathcal{I}_{\text{odd}} = \emptyset$; $\mathcal{I}_{\text{even}} = \emptyset$;
4 **for** $k = 1 \ldots d$,
5 compute λ_{2k-1} and λ_{2k} by using (21);
6 **if** $\lambda_{2k-1} < l_k$,
7 define $\mathcal{H}_{2k-1}$ as in (18);
8 $\mathcal{P}_{2k-1} = \mathcal{P} \cap \mathcal{H}_{2k-1}$;
9 $\mathcal{I}_{\text{odd}} = \mathcal{I}_{\text{odd}} \cup \text{multi-inner}(\mathcal{P}_{2k-1})$;
10 **if** $\lambda_{2k} > u_k$,
11 define $\mathcal{H}_{2k}$ as in (19);
12 $\mathcal{P}_{2k} = \mathcal{P} \cap \mathcal{H}_{2k}$;
13 $\mathcal{I}_{\text{even}} = \mathcal{I}_{\text{even}} \cup \text{multi-inner}(\mathcal{P}_{2k})$;
14 **return** $\mathcal{I}_{\text{odd}} \cup \mathcal{I}_{\text{even}} \cup \{\mathcal{B}\}$;
15 **else return** $\emptyset$.

Here, the function single-inner() computes the inner box according to any one of the methods proposed in Section 2. Note that Step 1 requires r before calling Algorithm 2, or x_0 before calling Algorithm 3. In the first case, either r is computed at each iteration by solving the linear programs (7) or (17), or is computed only once (at the first step) and kept constant. In the latter case, getting an $x_0 \in \mathcal{P}$ also requires the solution of a linear program only at the very first call. Indeed, for any $k = 1, 2, \ldots, n$, if $\mathcal{P}_{2k-1}$ is full-dimensional, an interior point is $x^{2k-1} = [x_j^{2k-1}]$ defined as

$$x_j^{2k-1} = \begin{cases} l_k + \frac{1}{2}\lambda_{2k-1} & \text{if } j = k, \\ \frac{1}{2}(u_j + l_j) & \text{otherwise.} \end{cases} \tag{23}$$

Analogously, for any $k = 1, 2, \ldots, n$, if $\mathcal{P}_{2k}$ is full-dimensional, an interior point is $x^{2k} = [x_j^{2k}]$ defined as

$$x_j^{2k} = \begin{cases} u_k + \frac{1}{2}\lambda_{2k} & \text{if } j = k, \\ \frac{1}{2}(u_j + l_j) & \text{otherwise.} \end{cases} \tag{24}$$

It should be noted that the representation of $\mathcal{P}_s$ ($s = 1, 2, \ldots, 2d$) differs from the given representation of $\mathcal{P}$ only for tighter bounds on some variables. Hence, if Algorithm 5 is used to compute a single inner box approximation of $\mathcal{P}_s$, then the application of a dual simplex method to optimize (7) or (17) starting from the already available basic solution of $\mathcal{P}$ seems the best choice.

Proposition 9. *The total number of recursive calls of Algorithm* 5 *is bounded by* $2d\lfloor \mathrm{vol}(\mathcal{P})/\varepsilon \rfloor$.

Proof. If a node in the recursive tree is not a leaf, then it corresponds to a box contained in $\mathcal{P}$ of volume greater than ε. There cannot be more than $\lfloor \mathrm{vol}(\mathcal{P})/\varepsilon \rfloor$ of such boxes with non-overlapping interiors. Since each node of the recursive tree may generate no more than $2d$ nodes, the preposition easily follows. □

The overall complexity of Algorithm 5 depends on the type of approximation computed at each call (cf. Section 2).

The following asymptotic property will be proved in Appendix A.

Theorem 2. *Let* $\mathcal{P} \subset \mathbb{R}^d$ *be a polytope and let* $\mathcal{I}_\varepsilon = \{\mathcal{B}_t\}_{t=1}^{S(\varepsilon)}$ *be its inner approximation generated by Algorithm* 5 *for a given* $\varepsilon > 0$ *when Algorithm* 2 *with* $r = e$ *is used for computing single inner approximations. Then*

$$\lim_{\varepsilon \to 0} \overline{\mathcal{I}_\varepsilon} \underset{a.e.}{=} \mathcal{P},$$

i.e., the Lebesgue measure of the difference $\mathcal{P} \setminus \overline{\mathcal{I}_\varepsilon}$ *tends to zero as* $\varepsilon \to 0$, *where*

$$\overline{\mathcal{I}_\varepsilon} = \bigcup_{t=1}^{S(\varepsilon)} \mathcal{B}_t \subseteq \mathcal{P}. \tag{25}$$

3.3. A recursive algorithm for outer approximation

In order to refine the outer approximation, given the minimum volume outer box of a full-dimensional polytope, we want to find a set of non-overlapping full-dimensional boxes whose union contains the polytope, each box having volume not greater than ε.

The following simple recursive Algorithm 6 performs such a task. As long as the current outer box has volume greater than ε, the box is divided into two equal parts by an hyperplane perpendicular to the longest edge (see Fig. 3). Then an outer box is computed for both the intersections of the polytope with the two parts.

Algorithm 6.

function $\mathcal{E} = \text{multi-outer}(\mathcal{P})$
1 $\mathcal{B}(l, u) = \text{single-outer}(\mathcal{P})$;

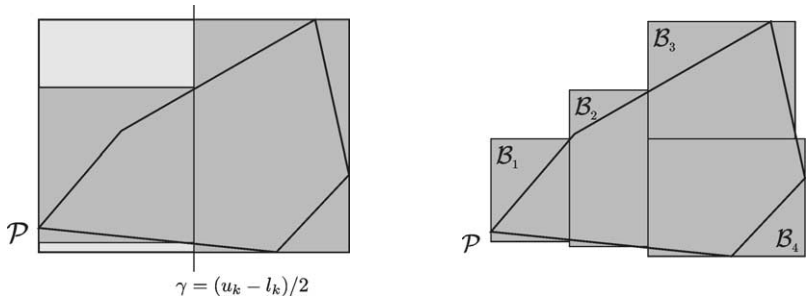


Fig. 3. Recursive outer approximation of a polytope $\mathcal{P}$ via Algorithm 6. (a) Approximation after the first recursion of Algorithm 6. (b) Result after two recursions.

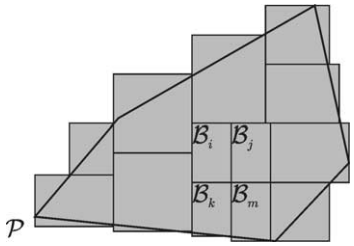


Fig. 4. Fragmentation of the outer approximation.

2 **if** $((\mathrm{vol}(\mathcal{B}) > \varepsilon)$,
3 **let** $k = \arg\max\{u_j - l_j\}$, $\gamma = (u_k - l_k)/2$;
4 $\mathcal{Q}_1 = \mathcal{P} \cap \{x \in \mathbb{R}^d\colon\ x_k \leqslant \gamma\}$;
5 $\mathcal{Q}_2 = \mathcal{P} \cap \{x \in \mathbb{R}^d\colon\ x_k \geqslant \gamma\}$;
6 **return** multi-outer$(\mathcal{Q}_1) \cup$ multi-outer$(\mathcal{Q}_2)$;
7 **else return** $\{\mathcal{B}(l, u)\}$.

The following observation allows to efficiently compute the outer boxes. Let $\mathcal{B}(l, u)$ be the minimum volume outer box of $\mathcal{P}$, and for each $j \in D$ let $x^{\min,j}$ and $x^{\max,j}$ be the optimal solutions of (17a) and (17b) respectively. Assume $\mathcal{B}(l, u)$ is divided in two equal parts along the kth coordinate. Accordingly, $\mathcal{P}$ is divided in two parts, $\mathcal{Q}_1 = \mathcal{P} \cap \{x \in \mathbb{R}^d\colon\ x_k \leqslant \gamma\}$ and $\mathcal{Q}_2 = \mathcal{P} \cap \{x \in \mathbb{R}^d\colon\ x_k \geqslant \gamma\}$, where $\gamma = (u_k - l_k)/2$. Let $\mathcal{B}(l^h, u^h)$ be the minimum volume outer box of $\mathcal{Q}_h$ $(h = 1, 2)$. Then the following statements are straightforward to prove for all $j \in D$:

$$x_k^{\min,j} \leqslant \gamma \quad \Rightarrow \quad l_j^1 = l_j, \qquad x_k^{\max,j} \leqslant \gamma \quad \Rightarrow \quad u_j^1 = u_j,$$
$$x_k^{\min,j} > \gamma \quad \Rightarrow \quad l_j^2 = l_j, \qquad x_k^{\max,j} > \gamma \quad \Rightarrow \quad u_j^2 = u_j.$$

Whenever one of the above condition applies, a linear program can be avoided for the computation of the outer box. As the conditions are mutually exclusive and collectively exhaustive, we need to solve only $2d$ new linear programs at each recursion.

Algorithm 6 may divide some boxes without reducing the volume error, therefore causing the fragmentation of the outer approximation, as depicted in Fig. 4. Such a fragmentation could be simply

avoided by checking that the generated box is not contained in the initial polytope $\mathcal{P}$ before recursively calling Algorithm 6. Clearly this check would prevent the computation of the boxes $\mathcal{B}_j$ and $\mathcal{B}_i$ in Fig. 4. The number of regions could be further reduced by backtracking if some box is contained in $\mathcal{P}$, and dividing along the second longest direction: in this case the boxes $\mathcal{B}_i$ and $\mathcal{B}_k$ as well as $\mathcal{B}_j$ and $\mathcal{B}_m$ would not be divided.

Proposition 10. *Let V denote the volume of the minimum volume outer box of $\mathcal{P}$. The total number of nodes in the recursive tree of Algorithm* 6 *is bounded by $\lceil 4V/\varepsilon \rceil$.*

Proof. At each recursive call, the current outer box is halved. Hence, the volume of the next outer box is no more than half the volume of the current one. It follows that when we are at level n of the recursion tree, the volume of the current outer box is no more than $2^{-n}V$. This quantity is not greater than ε if and only if $n \geqslant \log_2(V/\varepsilon)$. It follows that the depth of the recursive tree is bounded by $\lceil \log_2(V/\varepsilon) \rceil$. Since the recursive tree is binary, the total number of its nodes is bounded by

$$2^{\lceil \log_2(V/\varepsilon)\rceil+1} \leqslant 2^{\log_2(V/\varepsilon)+2} = \frac{4V}{\varepsilon}. \qquad \square$$

The following asymptotic result, analogous to Theorem 2, will be proved in Appendix B.

Theorem 3. *Let $\mathcal{P} \subset \mathbb{R}^d$ be a polytope. Let $\mathcal{E}_\varepsilon = \{\mathcal{B}_t\}_{t=1}^{T(\varepsilon)}$ be the outer approximation generated by Algorithm* 6 *for a given $\varepsilon > 0$. Then*

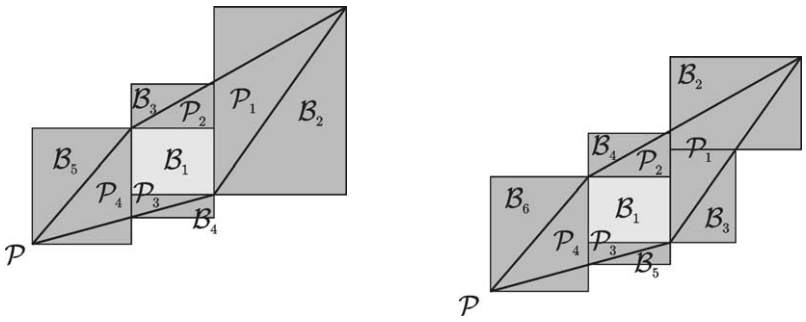
$$\lim_{\varepsilon \to 0} \overline{\mathcal{E}_\varepsilon} \underset{a.e.}{=} \mathcal{P},$$

where $\overline{\mathcal{E}_\varepsilon} = \bigcup_{t=1}^{T(\varepsilon)} \mathcal{B}_t$.

3.4. A recursive algorithm for inner and outer approximations

In this section we show how the results presented so far can be efficiently used to solve the problem stated in the introduction.

Given a polytope $\mathcal{P}$, consider the recursion tree generated by Algorithm 5. Each node corresponds to a polytope, which is generated either in step 7 or in step 11 of the algorithm, except for the root node, corresponding to the original $\mathcal{P}$. The leaves of the tree correspond to polytopes where the computed single inner box has volume not greater than ε (cf. step 2). Let $\mathcal{P}_\varepsilon^\ell$, with $\ell = 1, 2, \ldots, L(\varepsilon)$, denote the polytopes corresponding to the leaves of the recursion tree, for any fixed ε, and note that $(\overline{\mathcal{I}_\varepsilon}) \cup (\bigcup_{\ell=1}^{L(\varepsilon)} \mathcal{P}_\varepsilon^\ell) = \mathcal{P}$, where $\overline{\mathcal{I}_\varepsilon}$ is defined in (25). Clearly, we can easily compute an outer approximation by applying Algorithm 4 to each polyhedron $\mathcal{P}_\varepsilon^\ell$. However, as Fig. 5(a) shows, the volume of the outer boxes (e.g. $\mathcal{B}_3$) can be considerably larger than the tolerance ε used for the inner approximation. Algorithm 6 solves this problem by further approximating the outer boxes if their volume exceeds the threshold ε. The following inner-outer approximation algorithm summarizes the ideas discussed above and returns two collections of polyhedra $\mathcal{I}$, $\mathcal{M}$ such that $\mathcal{I}$ is an inner approximation and $\mathcal{E} = \mathcal{I} \cup \mathcal{M}$ is an outer approximation of $\mathcal{P}$.



(a) (b)

Fig. 5. Recursive inner and outer approximation of a polytope $\mathcal{P}$. (a) Inner and outer approximation using Algorithm 4 to approximate the rests $\mathcal{P}_i$. (b) Inner and outer approximation using Algorithm 6 to approximate the rests $\mathcal{P}_i$.

Algorithm 7.

function $[\mathcal{I}, \mathcal{M}] = \text{multi-inner-outer}(\mathcal{P})$
1 $\mathcal{B} = \text{single-inner}(\mathcal{P})$;
2 **if** $\text{vol}(\mathcal{B}) > \varepsilon$,
3 $\mathcal{I}_{\text{odd}} = \emptyset$; $\mathcal{I}_{\text{even}} = \emptyset$; $\mathcal{M}_{\text{odd}} = \emptyset$; $\mathcal{M}_{\text{even}} = \emptyset$;
4 **for** $k = 1 \dots d$,
5 compute λ_{2k-1} and λ_{2k} by using (21);
6 **if** $\lambda_{2k-1} < l_k$,
7 define $\mathcal{H}_{2k-1}$ as in (18);
8 $\mathcal{P}_{2k-1} = \mathcal{P} \cap \mathcal{H}_{2k-1}$;
9 $[\mathcal{I}_{\text{odd}}, \mathcal{M}_{\text{odd}}] = [\mathcal{I}_{\text{odd}}, \mathcal{M}_{\text{odd}}] \cup \text{multi-inner-outer}(\mathcal{P}_{2k-1})$;
10 **if** $\lambda_{2k} > u_k$,
11 define $\mathcal{H}_{2k}$ as in (19);
12 $\mathcal{P}_{2k} = \mathcal{P} \cap \mathcal{H}_{2k}$;
13 $[\mathcal{I}_{\text{even}}, \mathcal{M}_{\text{even}}] = [\mathcal{I}_{\text{even}}, \mathcal{M}_{\text{even}}] \cup \text{multi-inner-outer}(\mathcal{P}_{2k})$;
14 **return** $[\mathcal{I}_{\text{odd}} \cup \mathcal{I}_{\text{even}}, \mathcal{M}_{\text{odd}} \cup \mathcal{M}_{\text{even}}]$;
15 **else return** $[\emptyset, \text{outer}(\mathcal{P})]$.

All the different inner and outer approximation algorithms presented earlier can be combined in several ways by replacing the opportune functions in step 1 (single-inner-nlp, single-inner-lp, single-inner-greedy) and in step 15 (single-outer, multi-outer). In practice, many optimization problems (17) may be avoided since their optimal value is implicit in the definition of $\mathcal{P}_{2k-1}$ and $\mathcal{P}_{2k}$. Indeed, by recalling that $\mathcal{P}_{2k-1} = \mathcal{P} \cap \{x \in \mathbb{R}^d\colon l_j \leqslant x_j \leqslant u_j \ (0 \leqslant j \leqslant k-1), x_k \leqslant l_k\}$, we see that when Algorithm 7 is called with input $\mathcal{P}_{2k-1}$, we already know l_j for all $j = 1, \dots, k-1$ and u_j for all $j = 1, \dots, k$. Thus, it is sufficient to solve (17a) for all $j = k, \dots, d$ and (17b) for all $j = k+1, \dots, d$. Analogously, when Algorithm 7 is called with input $\mathcal{P}_{2k}$, it suffices to solve (17a) for all $j = k+1, \dots, d$ and (17b) for all $j = k, \dots, d$.

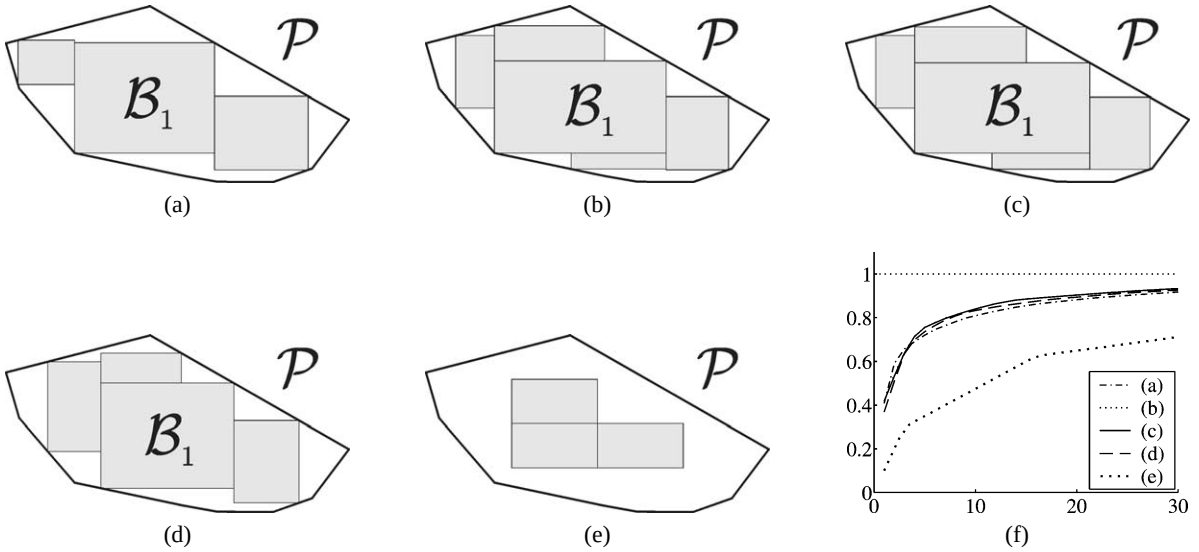


Fig. 6. Inner approximations. (a) Approximation of $\mathcal{P}$ using Algorithm 5 and performing the inner approximations with Algorithm 2 with $r = [1\ 1]'$. (b) Approximation of $\mathcal{P}$ using Algorithm 5 and performing the inner approximations with Algorithm 2 with r computed using (7) at each iteration. (c) Approximation of $\mathcal{P}$ using Algorithm 5 + Algorithm 2 with $r = u_h - l_h$ where $\mathcal{B}(l_h, u_h)$ is the outer box of $\mathcal{P}_h$, and where $\bigcup \mathcal{P}_h$ equals the closure of $\mathcal{P} \setminus \mathcal{B}_1$. (d) Approximation of $\mathcal{P}$ using Algorithm 5 and performing the inner approximations with Algorithm 3. (e) Approximation of $\mathcal{P}$ using Algorithm 2 presented in [4]. (f) Comparison of the approaches (cumulative volume against number of boxes), (a)–(e) as in Figs. 6(a)–(e).

4. Computational experience

This section presents a computational experience using the proposed algorithms. We first present a 2-dimensional polyhedron and its inner and outer approximations, then we present a statistical study of the inner and outer approximation algorithms on higher-dimensional objects.

4.1. Approximation of a 2-dimensional polytope

This section shows the same 2-polytope approximated using the presented algorithms. Fig. 6 reports the inner approximations. Note that the results depicted in Fig. 6(b) and in Fig. 6(c) are almost identical. Fig. 6(f) shows the cumulative sum of the first 30 boxes produced by each approximation, the constant line is the volume of the polytope computed using the package VINCI [3]. Fig. 7 shows the outer approximations of $\mathcal{P}$. Figs. 7(a) and 7(c) show that fragmentation happens only when Algorithm 6 is applied alone. Fig. 7(d) is computed by interpolation on 20 runs of the algorithms (termination volume ε between 1 and 0.05^2).

4.2. Higher dimensional approximations

In this section we consider the results obtained by running the algorithms on 100 random polytopes. Each input polytope is centered in the origin and is generated by uniformly distributing the normals of the facets of $\mathcal{P}$ and by randomly stretching and rotating the polytope. The polytopes have a unitary volume and the stopping criterion is set to $\varepsilon = 0.1^d$, where d is the dimension of the space embedding $\mathcal{P}$. Table 1

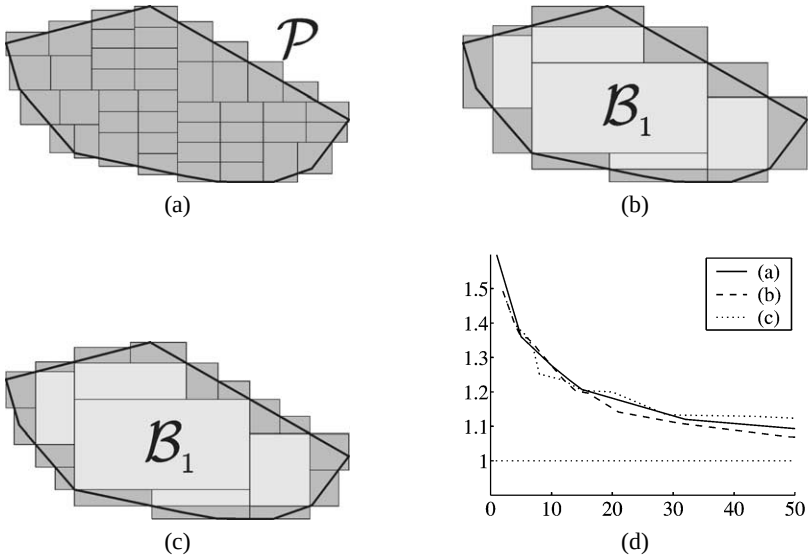


Fig. 7. Outer approximations. (a) Approximation of $\mathcal{P}$ using Algorithm 6. (b) Approximation of $\mathcal{P}$ using Algorithm 7 (Algorithm 2 with $r = u - l, \mathcal{E}(l,u) \supseteq \mathcal{P}$ and Algorithm 4). (c) Approximation of $\mathcal{P}$ using Algorithm 7 (Algorithm 2 with $r = u - l, \mathcal{E}(l,u) \supseteq \mathcal{P}$ and Algorithm 6). (d) Comparison of the approaches (cumulative volume against number of boxes) (a)–(c) as in Figs. 7(a)–(c).

Table 1
Average [variance] computational times (s) on a test pool of 100 polytopes, $\mathrm{vol}(\mathcal{P}) = 1$, $m = 2d$, $\varepsilon = 0.1^d$

Algorithm	$\mathcal{P} \in \mathbb{R}^2$	$\mathcal{P} \in \mathbb{R}^3$	$\mathcal{P} \in \mathbb{R}^4$
$5+2$ ($r=1$)	0.07 [0.02]	1.24 [0.14]	14.70 [1.70]
$5+2$ (r using (7))	0.25 [0.05]	5.51 [0.37]	104.50 [15.80]
$5+2$ ($r=u-l$)	0.18 [0.03]	3.18 [0.26]	56.54 [13.69]
$5+3$	0.03 [0.01]	1.17 [0.15]	18.98 [29.50]
2 in [4]	0.03 [0.01]	1.17 [0.15]	18.98 [29.50]
6	1.05 [0.08]	18.41 [0.98]	294.14 [15.67]
$7+2+4$	0.17 [0.03]	3.21 [0.28]	56.87 [13.69]
$7+2+6$	0.29 [0.04]	11.26 [2.27]	766.95 [389.60]

reports the computational times obtained by running a Matlab implementation of the approximation algorithms on a SUN workstation with a 950 MHz processor, using the LP solver E04MBF of the NAG Foundation Toolbox [12], while Table 2 and Table 3 report respectively the number of boxes and the volume of the approximation. Note that for higher dimensions d, even decreasing the tolerance $\varepsilon = 0.1^d$ the number of boxes remains constant, and the approximation error increases. This is a general consequence of approximating polytopes using simple shaped objects like boxes.

Note further that the performance of the algorithms rapidly deteriorates as the dimension grows. This fact is unavoidable, since the considered problem is inherently difficult. To support this point we observe that, though volume computation is not our primary concern, our inner-outer approximation algorithms return a lower and an upper bound of the volume of $\mathcal{P}$. In [1] it is proved that every polynomial time

Table 2
Average [variance] number of generated boxes on a test pool of 100 polytopes, $\mathrm{vol}(\mathcal{P}) = 1$, $m = 2d$, $\varepsilon = 0.1^d$

Algorithm	$\mathcal{P} \in \mathbb{R}^2$	$\mathcal{P} \in \mathbb{R}^3$	$\mathcal{P} \in \mathbb{R}^4$
$5+2$ ($r=1$)	11.07 [1.70]	96.43 [8.77]	631.97 [66.03]
$5+2$ (r using (7))	11.04 [1.58]	95.46 [4.24]	573.45 [54.10]
$5+2$ ($r=u-l$)	11.14 [1.64]	94.66 [5.03]	538.24 [69.64]
$5+3$	11.11 [1.45]	70.35 [5.23]	415.88 [31.98]
2 in [4]	11.97 [1.98]	226.32 [49.14]	843.01 [248.86]
6	159.35 [9.14]	1797.65 [79.17]	21728.56 [886.44]
$7+2+4$	30.99 [3.83]	362.46 [20.96]	2824.52 [333.51]
$7+2+6$	52.76 [5.28]	1061.10 [128.01]	18965.68 [1692.87]

Table 3
Average [variance] volume of the approximation on a test pool of 100 polytopes, $\mathrm{vol}(\mathcal{P}) = 1$, $m = 2d$, $\varepsilon = 0.1^d$

Algorithm	$\mathcal{P} \in \mathbb{R}^2$	$\mathcal{P} \in \mathbb{R}^3$	$\mathcal{P} \in \mathbb{R}^4$
$5+2$ ($r=1$)	0.82 [0.02]	0.59 [0.04]	0.35 [0.03]
$5+2$ (r using (7))	0.84 [0.02]	0.61 [0.05]	0.32 [0.05]
$5+2$ ($r=u-l$)	0.85 [0.02]	0.61 [0.05]	0.31 [0.05]
$5+3$	0.84 [0.02]	0.14 [0.20]	0.23 [0.03]
2 in [4]	0.54 [0.04]	0.51 [0.05]	0.28 [0.04]
6	1.07 [0.01]	1.21 [0.03]	1.43 [0.04]
$7+2+4$	1.16 [0.02]	1.56 [0.21]	3.38 [1.04]
$7+2+6$	1.08 [0.01]	1.21 [0.03]	1.48 [0.06]

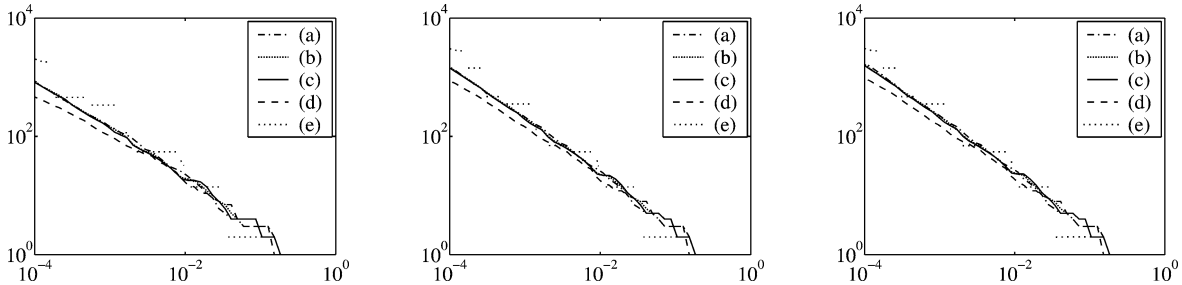
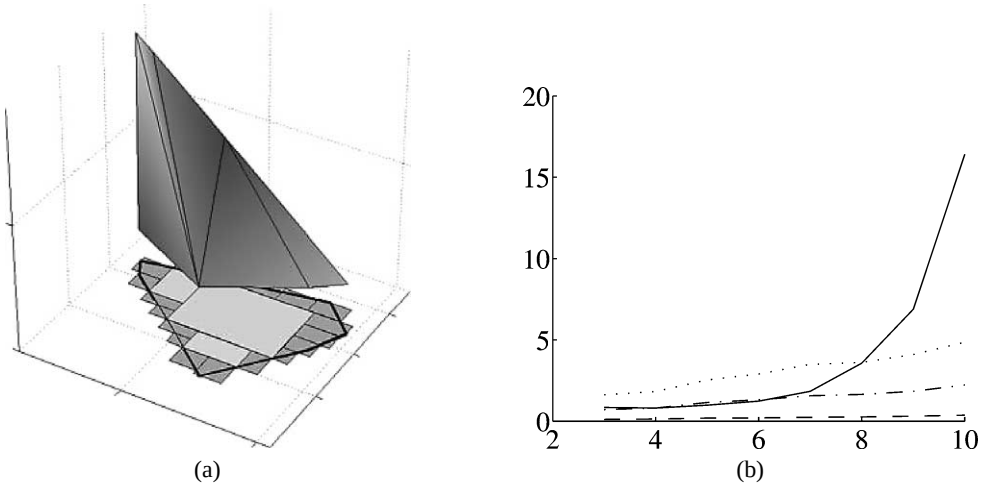


Fig. 8. Number of generated inner boxes against ε, average for 10 runs, $\mathrm{vol}(\mathcal{P}) = 1$, line-styles are associated to Algorithms as in Fig. 6. (a) $\mathcal{P} \in \mathbb{R}^3$. (b) $\mathcal{P} \in \mathbb{R}^4$. (c) $\mathcal{P} \in \mathbb{R}^5$.

algorithm computing a lower and an upper bound on the volume of a convex set commits a relative error which grows exponentially with the dimension.

The relation between the volume tolerance ε and the number of generated inner boxes is illustrated in Fig. 8, where the plots suggest an exponential relation among the two quantities.



(a) (b)

Fig. 9. Approximated projection on $\mathbb{R}^2$ of a polytope $\mathcal{P}$ using modified versions of Algorithm 5 (Algorithm 2 with $r = u - l, \mathcal{E}(l,u) \supseteq \mathcal{P}$) for the inner approximation and Algorithm 4 for the outer. (a) Approximate projection of a polytope $\mathcal{P} \in \mathbb{R}^3$. (b) Elapsed time (s) against dimension of $\mathcal{P}$, average for 10 runs ($\varepsilon = (0.5)^2$ dashed, $\varepsilon = (0.1)^2$ dash-dotted, $\varepsilon = (0.05)^2$ dotted, exact computation via [8] solid).

5. Extension to approximate projections

Consider the problem of projecting a polytope $\mathcal{P} \subset \mathbb{R}^\sigma$ onto the linear subspace L generated by the first d vectors of the canonical basis.[1] Then, projecting $\mathcal{P}$ onto L amounts to find a representation of $\{x \in \mathbb{R}^d \colon \exists \gamma \in \mathbb{R}^{\sigma-d}, \left[\begin{smallmatrix} x \\ \gamma \end{smallmatrix}\right] \in \mathcal{P}\}$ in terms of linear inequalities. If $\mathcal{P}$ is given by a set of linear inequalities, then the projection problem is a difficult one, which can be solved for instance by using the Fourier-Motzkin elimination method [17]. On the other hand, the algorithms proposed in this paper can be extended to directly compute the approximation of the projection of $\mathcal{P}$ as the union of d-dimensional boxes, short of computing the projection itself or the projection of the inner approximation of $\mathcal{P}$, as Fig. 9(a) shows for the projection on the first two coordinate axes of a 3-dimensional polytope.

Let A_D be the first d columns of A, and A_N such that $A = [A_D \; A_N]$. The maximum volume r-constrained inner box contained in the projection (cf. problem (5)) is $\mathcal{B}(x^*, x^* + \lambda^* r)$, where x^*, λ^* solve the LP

$$\begin{aligned} \max_{x,\gamma_S,\lambda} \quad & \lambda \\ \text{subject to} \quad & A_D\big(x + v(S) r\lambda\big) + A_N \gamma_S \leqslant b \quad (\forall S \subseteq D), \\ & x \in \mathbb{R}^d, \quad \lambda \in \mathbb{R}, \\ & \gamma_S \in \mathbb{R}^{\sigma-d} \quad (\forall S \subseteq D), \end{aligned} \tag{26}$$

where $v(S) \in \{0,1\}^d$ is the incidence vector of the subset of coordinate indices $S \subseteq D$, as defined in (1). Note that (26) contains $d + (\sigma - d)2^d + 1$ variables and $m2^d$ constraints. Fig. 9(b) summarizes the following computational experience: Given $\mathcal{P} \in \mathbb{R}^\sigma$, defined as the intersection of 2σ hyperplanes, compute the projection on the first two coordinate axes ($d = 2$). The tests were run with different

[1] In case L is a generic affine subspace, it is enough to perform a standard linear coordinate transformation.

tolerances ε for the inner and outer approximation and compared with the computational times needed for the exact projection using the package CDD [8]. While for small dimensions σ the two approaches have comparable timings, they scale differently when the dimension σ increases.[2]

On the other hand, because of the exponential explosion of variables and constraints with d in (26), when $\sigma \approx d$ problem (26) may be not convenient, compared to Fourier–Motzkin elimination. In this case, we propose to replace (26) with the LP

$$\begin{aligned} &\max_{x,\gamma,\lambda} && \lambda \\ &\text{subject to} && A_D x + A_D^+ r\lambda + A_N \gamma \leqslant b \\ &&& x \in \mathbb{R}^d, \quad \lambda \in \mathbb{R}, \quad \gamma \in \mathbb{R}^{\sigma-d}, \end{aligned} \tag{27}$$

where A_D^+ is the positive part of A_D, which returns the largest r-constrained inner box of dimension d contained in $\mathcal{P}$. Problem (27) has $\sigma + 1$ variables and m constraints, however it returns a box which, in general, has a smaller volume than the one provided by (26).

We conclude by assessing the impact of the technique proposed here on the problem that originally motivated our interest. As recalled in the introduction, the reach set computation determines the set of configurations (or *states*) that a system can reach. Let us restrict our attention to a piecewise linear system $x(k+1) = A_j x(k) + B_j u(k)$, where $x(k) \in \mathbb{R}^n$ is the *state*, $u(k) \in \mathbb{R}^m$ is the *input* to the system[3], A_j, B_j are matrices of suitable dimensions, and $j \in \{1, \ldots, s\}$ is the current "mode". Assuming that the initial state $x(0) \in \mathcal{X}$, and that $u(k) \in \mathcal{U}$ for $k \geqslant 0$, where $\mathcal{X}$ and $\mathcal{U}$ are polytopes, then it is possible to compute the set of states $\text{Reach}(\mathcal{X}, \mathcal{U}, K)$ that are reachable in K steps from any initial condition in $\mathcal{X}$ and subject to any input in $\mathcal{U}$ as

$$\begin{aligned} \text{Reach}(\mathcal{X}, \mathcal{U}, K) = \big\{ y\colon\ & y = A_j^K x(0) + A_j^{K-1} B_j u(0) + A_j^{K-2} B_j u(1) + \cdots + B_j u(K-1), \\ & x(0) \in \mathcal{X},\ u(i) \in \mathcal{U},\ i = 0, \ldots, K-1 \big\}. \end{aligned} \tag{28}$$

Clearly (28) defines the projection of a polytope of dimension $(Km + n)$ onto the subspace of dimension n spanned by the rows of the matrix $[A_j^K, A_j^{K-1} B_j, A_j^{K-2} B_j, \ldots, B_j]$. The advantages that the proposed technique brings in the reach set computations are a consequence of the good performance of the approximate computation of the projection, and are more evident when Km increases. Note that, during the reachability analysis of hybrid systems the projection (28) is performed several times for different sets $\mathcal{X}$, modes $j \in \{1, \ldots, s\}$ and horizons K. In particular, the value of K is related to the time the hybrid system remains in a certain mode, and may vary considerably depending on the system dynamics under analysis [15].

6. Conclusions

This paper presents a collection of algorithms to compute an inner and an outer approximation of a given polytope. Such algorithms face the multiple objective problem of minimizing both the volume error

[2] The absolute times reported are relative to a SUN workstation with a 950 MHz processor running a Matlab implementation of the approximation algorithms and the C implementation of CDD.

[3] Here we prefer to follow the standard control engineering notation even if it may conflict with the notation adopted in the rest of the paper, the context should disambiguate the symbols.

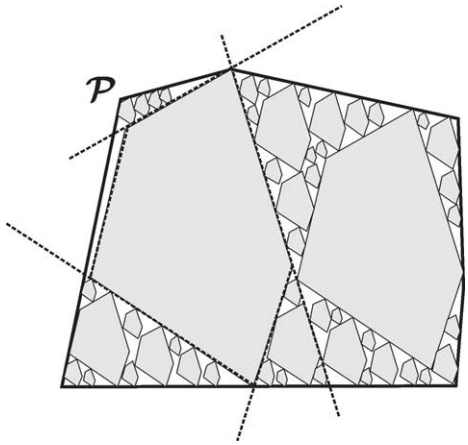


Fig. 10. Approximation of a polytope by using generic polytopic shapes.

and the number of boxes needed for the approximation. The algorithms with minor modifications are a computational attractive alternative to the exact computation of the projection.

Although our techniques were conceived and used to improve the performance of the reachability analysis algorithm [15], they could be used successfully in other applicative domains. The performance are good as the computational experiences suggest.

Extensions of the approach include the approximation of generic convex sets by hyper-boxes, and the generalization of the inner approximation algorithm to generic polytopal shapes (see the experiment shown in Fig. 10, where a pentagonal shape is used as approximating shape, and recursiveness is based on the partition of polytope $\mathcal{P}$ by means of linear cuts generated by the facet inequalities of the largest approximating inner polytope). In both cases, we expect almost-everywhere convergence to $\mathcal{P}$ of the resulting inner and outer approximations as $\varepsilon \to 0$.

Moreover, it is a research topic under investigation how to determine the projection of $\mathcal{P}$ on a subspace L (or a polyhedral approximation of such a projection related to ε) by efficiently using the information provided by the inner and outer multi-rectangular approximation of the projection itself.

Appendix A. Proof of Theorem 2

If $\mathcal{P}$ is not full dimensional, $0 = \mathrm{vol}(\mathcal{P}) = \mathrm{vol}(\mathcal{I}_\varepsilon)$ for all $\varepsilon > 0$. Hence, assume $\mathcal{P}$ is a full dimensional set. Clearly, the function $\mathrm{vol}(\mathcal{I}_\varepsilon)$ is nonnegative and monotonically increasing as $\varepsilon \to 0$, and $0 \leqslant \lim_{\varepsilon\to 0} \mathrm{vol}(\mathcal{I}_\varepsilon) = \underline{m} \leqslant \mathrm{vol}(\mathcal{P})$. By contradiction, assume $\underline{m} < \mathrm{vol}(\mathcal{P})$. Then, there exist a point $\bar{x} \in \mathcal{P}$ and a scalar $\sigma > 0$ such that the hypercube $\mathcal{Z} = \mathcal{B}(\bar{x} - \sigma e, \bar{x} + \sigma e)$ is strictly contained in $\mathcal{P}$ and

$$\mathcal{Z} \cap \mathcal{I}_\varepsilon = \emptyset, \tag{A.1}$$

for all $\varepsilon > 0$.

Consider the recursion tree generated by Algorithm 5. Each node corresponds to a polytope, which is generated either in step 8 or in step 12 of the algorithm, except for the root node, corresponding to the original $\mathcal{P}$. The leaves of the tree correspond to polytopes where the computed single inner box has volume less than ε (cf. step 2). Let $\mathcal{P}_\varepsilon^t$, with $t = 1, 2, \ldots, \tau(\varepsilon)$, denote the polytopes corresponding to

the leaves of the recursion tree, for any fixed ε. Since $\bigcup_{t=1}^{\tau(\varepsilon)} \mathcal{P}_\varepsilon^t$ equals the closure of $\mathcal{P} \setminus \mathcal{I}_\varepsilon$, we have $\mathcal{Z} \subseteq \bigcup_{t=1}^{\tau(\varepsilon)} \mathcal{P}_\varepsilon^t$. Let $\mathcal{D}_t = \mathcal{Z} \cap \mathcal{P}_\varepsilon^t$, for all $t = 1, 2, \ldots, \tau(\varepsilon)$. Since $\mathcal{Z}$ is contained in the interior of $\mathcal{P}$, all the sets $\mathcal{D}_t$ with a nonempty interior are boxes, as their hyperplane representation can only contain facet inequalities of $\mathcal{Z}$ and hyperplanes generated according by Algorithm 5. To simplify the notation, in the following we denote by $\mathcal{R}$ the generic $\mathcal{P}_\varepsilon^t$ and by $\mathcal{D}$ the corresponding box $\mathcal{D}_t$, and only consider sets $\mathcal{D}_t$ having a nonempty interior. Furthermore we set

$$\mathcal{D} = \{\alpha_k \leqslant x_k \leqslant \alpha_k + \rho_k \ (k = 1, \ldots, d)\},$$
$$J = \{k\text{: both } \alpha_k \leqslant x_k \text{ and } x_k \leqslant \alpha_k + \rho_k \text{ are facet inequalities of } \mathcal{R} \ (k = 1, \ldots, d)\},$$
$$\underline{\rho} = \min\{\rho_k\text{: } k = 1, \ldots, d\},$$
$$\overline{\rho} = \min\{\rho_k\text{: } k \in J\}.$$

Note that J might be empty, in that case we set $\overline{\rho} = \infty$. Finally, let λ denote the edge length of a maximum volume inner hypercube of $\mathcal{R}$.

Note that $\underline{\rho} \leqslant \lambda \leqslant \overline{\rho}$, and that if $\underline{\rho} = \overline{\rho}$ then the edge length of the maximum volume hypercube is equal to the common value. The following claim is a consequence of the partition procedure of Algorithm 5.

Claim A.1. *If $\underline{\rho} = \overline{\rho}$ then $\mathcal{D}$ is not cut orthogonally to the kth coordinate axis, for all $k \in J$ in further recursions of the algorithm.*

Claim A.2. *If* (A.1) *holds for some $\bar{\varepsilon} > 0$ then every $\mathcal{D}$ corresponding to $\bar{\varepsilon}$ is subdivided in further recursions of the algorithm for ε sufficiently small, with $\bar{\varepsilon} > \varepsilon > 0$.*

Proof of Claim A.2. When Algorithm 5 is applied to $\mathcal{R}$ with $\varepsilon \leqslant \underline{\rho}$, a hypercube $\mathcal{B} \subseteq \mathcal{R}$ is found with $\operatorname{vol}(\mathcal{B}) \geqslant \underline{\rho}^d$. Exactly one of the following three cases can occur:

(1) $\mathcal{D}$ is cut;
(2) $\mathcal{B} \cap \mathcal{D} \neq \emptyset$;
(3) in the next recursive call, $\mathcal{D}$ is contained in a region $\mathcal{R}'$ such that:
  (i) $\operatorname{vol}(\mathcal{R}') \leqslant \operatorname{vol}(\mathcal{R}) - \operatorname{vol}(\mathcal{B}) \leqslant \operatorname{vol}(\mathcal{R}) - \underline{\rho}^d$;
  (ii) $\mathcal{R}'$ contains a hypercube $\mathcal{B}'$ with $\operatorname{vol}(\mathcal{B}') \geqslant \underline{\rho}^d$.

Since $\mathcal{B} \subseteq \mathcal{I}_\varepsilon$, case (2) contradicts (A.1), and thus must be excluded. Case (3) can happen only a finite number of times. Hence, for $\varepsilon \leqslant \underline{\rho}$, Algorithm 5 recursively cuts $\mathcal{D}$ in smaller boxes. □

Note that at least one facet inequality of each $\mathcal{D}$ must be a facet inequality of $\mathcal{Z}$, since otherwise the corresponding rest polytope $\mathcal{R}$ is contained in $\mathcal{Z}$.

Claim A.3. *For $\varepsilon > 0$ sufficiently small, there exists an $\mathcal{R}$ and a corresponding $\mathcal{D} = \mathcal{R} \cap \mathcal{Z}$ for which there is an $r \in \{1, \ldots, d\}$ such that*

(i) $\rho_k = \underline{\rho}$ *for all* $k \neq r$;
(ii) *for all* $k \neq r$, *both* $\alpha_k \leqslant x_k$ *and* $x_k \leqslant \alpha_k + \underline{\rho}$ *are facet inequalities of* $\mathcal{R}$;
(iii) $\rho_r > \underline{\rho}$.

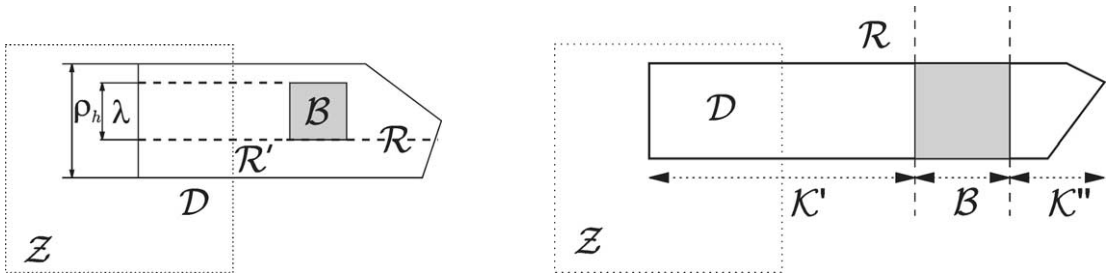


Fig. A.1. Proof of Theorem 2. (a) Exemplification of subcase 2.1 by a section in $\mathbb{R}^2$. (b) Exemplification in $\mathbb{R}^2$ of the situation considered in Claim 4.

Proof of Claim A.3. Assume that, for some $\bar{\varepsilon} > 0$, there is no box $\mathcal{D}$ satisfying the conditions of Claim A.3. By Claim A.2, for $\varepsilon \leqslant \bar{\varepsilon}$ sufficiently small, the same $\mathcal{D}$ is cut by one or more hyperplanes. Let $\mathcal{B} = \{\gamma_k \leqslant x_k \leqslant \gamma_k + \lambda \ (k = 1, \ldots, d)\}$ be the current maximum volume inner hypercube. Then the cutting hyperplanes are of the form $x_h = \gamma_h$ and $x_h = \gamma_h + \lambda$.

We distinguish between two cases:

Case 1 There is a cutting hyperplane $x_h = \beta$ such that at least one of $\alpha_h \leqslant x_h$ and $x_h \leqslant \alpha_h + \rho_h$ is a facet inequality of $\mathcal{Z}$. Then at least one piece $\mathcal{D}'$ of $\mathcal{D}$ has one more facet inequality of the corresponding rest $\mathcal{R}'$.

Case 2 If $x_h = \beta$ is a cutting hyperplane then both $\alpha_h \leqslant x_h$ and $x_h \leqslant \alpha_h + \rho_h$ are facet inequalities of $\mathcal{R}$. Then for any pair $x_h = \gamma_h$ and $x_h = \gamma_h + \lambda$ of cutting hyperplanes we have $\alpha_h \leqslant \gamma_h$ and $\gamma_h + \lambda \leqslant \alpha_h + \rho_h$. We have to distinguish between two further subcases:

Subcase 2.1: $\underline{\rho} < \overline{\rho}$. Note that if we fix two points with distance $\lambda > 0$ on a line segment of length $\rho_h > \lambda$ then the segment is divided in three or two parts, and at least one of them has length not greater than $\rho_h/2$. As a consequence, we can find a full dimensional piece $\mathcal{D}'$ of $\mathcal{D}$ with at least one edge corresponding to opposite facet inequalities of $\mathcal{R}' \subset \mathcal{R}$ having length at most one half of the length of the corresponding edge of $\mathcal{D}$, see Fig. A.1(a).

Subcase 2.2: $\underline{\rho} = \overline{\rho}$. In this case $\lambda = \underline{\rho}$. If we take as $\mathcal{D}'$ the full dimensional piece of $\mathcal{D}$ contained between every pair of cutting hyperplanes then we have at least one more pair of facet inequalities of $\mathcal{R}'$ whose distance is $\underline{\rho}$; furthermore, $\underline{\rho}' = \underline{\rho}$.

Since at most $2d - 1$ facet inequalities of $\mathcal{D}'$ can be facet inequalities of $\mathcal{R}'$, case 1 can happen a finite number of times. Since $\underline{\rho} > 0$, subcase 2.1 can happen consecutively a finite number of times before we get $\underline{\rho} = \overline{\rho}$. (More precisely, one can argue that the number of consecutive repetitions of subcase 2.1 is bounded by $\sum_{k \in J} \lceil \log_2 \rho_k - \log_2 \underline{\rho} \rceil$.) When subcase 2.2 is entered, it can be repeated at most $d - 1$ consecutive times before all pairs of facet inequalities of $\mathcal{R}$ have distance $\underline{\rho}$. When this happens, either all pairs of facet inequalities of $\mathcal{D}$ but one are pairs of facet inequalities of $\mathcal{R}$ or not. In the former case, $\mathcal{D}$ satisfies the conditions of the claim and we are done. In the latter case by Claim A.2, $\mathcal{D}$ has to be cut further, and by Claim A.1 this must necessarily occur orthogonally to a direction such that at least one facet inequality of $\mathcal{D}$ is not a facet inequality of $\mathcal{R}$, hence falling in case 1. □

Claim A.4. *Let $\mathcal{R}$ and $\mathcal{D}$ satisfy the conditions of Claim* A.3*, and let ε be such that $0 < \varepsilon \leqslant \underline{\rho}$. Then Algorithm* 5 *finds a hypercube $\overline{\mathcal{B}}$ such that $\overline{\mathcal{B}} \cap \mathcal{D} \neq \emptyset$.*

Proof of Claim A.4. When Algorithm 5 is run on $\mathcal{R}$ with $\varepsilon = \underline{\rho}$ then a hypercube $\mathcal{B}$ with edge length $\underline{\rho}$ is found.

If $\mathcal{B} \cap \mathcal{D} \neq \emptyset$ then by setting $\overline{\mathcal{B}} = \mathcal{B}$ we prove the claim.

If $\mathcal{B} \cap \mathcal{D} = \emptyset$ then $\mathcal{R}$ is divided in three parts: $\mathcal{K}' = \mathcal{R} \cap \{x_r \leqslant \beta_r\}$, $\mathcal{B} = \{\beta_k \leqslant x_k \leqslant \beta_k + \underline{\rho}\ (k = 1, \ldots, d)\}$, $\mathcal{K}'' = \mathcal{R} \cap \{x_r \geqslant \beta_r + \underline{\rho}\}$. Assume without loss of generality that $\mathcal{D} \subseteq \mathcal{K}'$ (see Fig. A.1(b)), and note that $\mathcal{K}'$ is a box which may be written as

$$\mathcal{K}' = \left\{\alpha_r \leqslant x_r \leqslant \beta_r, \alpha_k \leqslant x_k \leqslant \alpha_k + \underline{\rho}\ (k \neq r)\right\},$$

where $\beta_r - \alpha_r \geqslant \rho_r > \underline{\rho}$. When Algorithm 5 is applied to $\mathcal{K}'$ with $\varepsilon \leqslant \underline{\rho}$, a hypercube $\mathcal{B} \subseteq \mathcal{K}'$ is found with $\mathrm{vol}(\mathcal{B}) = \underline{\rho}^d$. Exactly one of the following two cases occurs:

(1) $\mathcal{B} \cap \mathcal{D} \neq \emptyset$;
(2) in the next recursive call, $\mathcal{D}$ is contained in a region $\mathcal{K}'''$ such that
  (i) $\mathrm{vol}(\mathcal{K}''') \leqslant \mathrm{vol}(\mathcal{K}') - \mathrm{vol}(\mathcal{B}) = \mathrm{vol}(\mathcal{K}') - \underline{\rho}^d$;
  (ii) $\mathcal{K}'''$ contains a hypercube $\mathcal{B}'''$ with $\mathrm{vol}(\mathcal{B}''') = \underline{\rho}^d$.

Since case (2) can happen only a finite number of times, case (1) must be met. When this happens, by setting $\overline{\mathcal{B}} = \mathcal{B}$ we prove the claim. □

As a consequence of the above claims, if Algorithm 5 is run on $\mathcal{P}$ for $\varepsilon > 0$ sufficiently small, in a finite number of steps a hypercube $\overline{\mathcal{B}}$ contained in $\mathcal{I}_\varepsilon$ with positive measure is met, such that $\overline{\mathcal{B}} \cap \mathcal{Z} \neq \emptyset$, contradicting (A.1). □

Appendix B. Proof of Theorem 3

Lemma B.1. *Let $\mathcal{P} \subset \mathbb{R}^d$ be a polytope. Let $\mathcal{E}_\varepsilon = \{\mathcal{B}_t\}_{t=1}^{T(\varepsilon)}$ be the outer approximation generated by Algorithm* 6 *for a given $\varepsilon > 0$. Then*

$$\lim_{\varepsilon \to 0} \max_{t \in \{1,\ldots,T(\varepsilon)\}} \left\{|u_t - l_t|_\infty\right\} = 0.$$

Proof. The limit exists because $\phi(\varepsilon) = \max_{t \in \{1,\ldots,T(\varepsilon)\}}\{|u_t - l_t|_\infty\}$ is the length of the longest edge of the outer approximation $\{\mathcal{B}_t\}_{t=1}^{T(\varepsilon)}$ and $\phi(\varepsilon)$ is monotonically non-increasing for $\varepsilon \to 0$. The limit is 0 because if by contradiction it were not, then it would exist an edge length λ such that $|u_t - l_t|_\infty \geqslant \lambda$, $\forall t, \varepsilon$, and $\mathrm{vol}(\mathcal{B}_t) \leqslant \varepsilon$, therefore the volume reduction would be achieved by preserving the length of the longest edge, and this would contradict the formulation of Algorithm 6. □

Proof of Theorem 3. We directly apply the definition of limit, therefore we want to show that for every z, exists $\bar{\varepsilon}$ such that if $\varepsilon < \bar{\varepsilon}$ then the following holds: $z \in \mathcal{P} \Leftrightarrow z \in \bigcup_{t=1}^{T(\varepsilon)} \mathcal{B}_t$. Clearly, if $z \in \mathcal{P}$, then $z \in \bigcup_{t=1}^{T(\varepsilon)} \mathcal{B}_t$, because by construction the set $\mathcal{P} \subseteq \bigcup_{t=1}^{T(\varepsilon)} \mathcal{B}_t$. In order to find $\bar{\varepsilon}$ such that if $\varepsilon \leqslant \bar{\varepsilon}$,

then $z \in \mathcal{P} \Leftarrow z \in \bigcup_{t=1}^{T(\varepsilon)} \mathcal{B}_t$, we assume by contradiction that $z \notin \mathcal{P}$. Let us define the distance between z and the set $\mathcal{P}$ as

$$\lambda = \min_x |x - z|_\infty \quad \text{subject to } x \in \mathcal{P}, \tag{B.1}$$

where clearly $\lambda > 0$ because $z \notin \mathcal{P}$. By Lemma B.1, given λ it exists $\tilde{\varepsilon}$ such that if $\varepsilon < \tilde{\varepsilon}$, then $|u_t - l_t|_\infty < \lambda$ for all $\mathcal{B}_t(l_t, u_t)$. By setting $\bar{\varepsilon} = \tilde{\varepsilon}$, we get that $z \notin \bigcup_{t=1}^{T(\varepsilon)} \mathcal{B}_t$ because otherwise the box containing z would not intersect $\mathcal{P}$. □

References

[1] I. Bárány, Z. Füredi, Computing the volume is difficult, Discrete Comput. Geom. 2 (1987) 319–326.
[2] A. Bemporad, K. Fukuda, F.D. Torrisi, Convexity recognition of the union of polyhedra, Computational Geometry 18 (2001) 141–154.
[3] B. Büeler, A. Enge, K. Fukuda, Exact volume computation for convex polytopes: A practical study, in: G. Kalai, G. Ziegler (Eds.), Polytopes, Combinatorics and Computation, in: DMV-Seminar, vol. 29, Birkhauser Verlag, 2000.
[4] J. Cohen, T. Hickey, Two algorithms for determining volumes of convex polyhedra, J. ACM 26 (3) (1979) 401–414.
[5] B.C. Eaves, R.M. Freund, Optimal scaling of balls and polyhedra, Math. Programming 23 (1982) 138–147.
[6] R.M. Freund, J.B. Orlin, On the complexity of four polyhedral set containment problems, Math. Programming 33 (1985) 139–145.
[7] K. Fukuda, Polyhedral computation FAQ, Both html and ps versions available from http://www.ifor.math.ethz.ch/~fukuda/fukuda.html.
[8] K. Fukuda, cdd/cdd+ Reference Manual, Institute for Operations Research ETH-Zentrum, http://www.ifor.math.ethz.ch/~fukuda/cdd_home/cdd.html, 0.61 (cdd) 0.76 (cdd+) edition, December 1997.
[9] J.E. Goodman, Joseph O'Rourke (Eds.), Handbook of Discrete and Computational Geometry, Discrete Mathematics and Its Applications, CRC Press, New York, 1997.
[10] P. Gritzmann, V. Klee, Computational complexity of inner and outer j-radii of polytopes in finite-dimensional normed spaces, Math. Programming 59 (1993) 163–213.
[11] P. Gritzmann, V. Klee, On the complexity of some basic problems in computational convexity: I. Containment problems, Discrete Math. 136 (1994) 129–174.
[12] NAG Numerical Algorithms Group Ltd., The MathWorks Inc., http://www.nag.com, NAG Foundation Toolbox—For Use with MATLAB, 1995.
[13] Y. Nesterov, A. Nemirovskii, Interior-Point Polynomial Algorithms in Convex Programming, Studies in Applied Mathematics, SIAM, Philadelphia, 1994.
[14] J.-R. Sack, J. Urrutia (Eds.), Handbook of Computational Geometry, North-Holland, Amsterdam, 2000.
[15] F.D. Torrisi, A. Bemporad, Discrete-time hybrid modeling and verification, in: Proc. 40th IEEE Conf. on Decision and Control, 2001, pp. 2899–2904.
[16] B. Zhu, Approximating convex polyhedra with axis-parallel boxes, Internat. J. Comput. Geom. Appl. 7 (3) (1997) 253–267.
[17] G.M. Ziegler, Lectures on Polytopes, Springer-Verlag, Berlin, 1995.